Technische Universität München
TUM School of Engineering and Design

Selected Aspects of Technical Debt in Production Automation

Quang Huan Dong

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Markus Zimmermann

Prüfer*innen der Dissertation:

1. Prof. Dr.-Ing. Birgit Vogel-Heuser
2. Prof. Dr. Matthias Tichy

Die Dissertation wurde am 09.02.2023 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 25.04.2023 angenommen.

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Birgit Vogel-Heuser, for the support and advice throughout my studies. I am fortunate to have a supervisor who enthusiastically helped me whenever I had questions.

Moreover, I would like to thank Professor Matthias Tichy for the mentorship. Also, I would like to thank Professor Markus Zimmermann for chairing the examination committee.

Furthermore, I would like to thank the Vietnam International Education Development and the Vietnamese-German University for granting me a scholarship under Project 911 - "Training lecturers of Doctor's Degree for universities and colleges for the 2010-2020 period" (Decision No. 771/QD-BGDDT dated 14/03/2017), which made my studies possible.

In addition, I want to thank my colleagues at the Institute of Automation and Information Systems, Technical University of Munich, for their support and advice.

Last but not least, I want to take this opportunity to thank my family, relatives and friends for their support and encouragement.

Acronyms

acatech	German National Academy of Science and Engineering
AIS	Institute of Automation and Information Systems
aPS	automated production systems
BPMN	Business Process Model and Notation
CLD	Causal Loop Diagram
E/E	electrical/electronics
GAMP	Good Automated Manufacturing Practice
MM	machine manufacturers
OMAC	Organization for Machine Automation and Control
PLC	Programmable Logic Controller
PM	plant manufacturers
POU	Program Organization Unit
RA	research activity
RBP	Reference Behaviour Patterns
Req	requirement
RPI4DD	Risk Prioritisation Indicator of Documentation Debt
SWMAT4aPS	Software Maturity for aPS
TD	Technical Debt
TD4aPS	Technical Debt for aPS
TUM	Technical University of Munich

Table of Contents

1. Introduction

Coined by Cunningham (1993) in the classical software engineering domain, the Technical Debt (TD) metaphor refers to a context in which a technical compromise is selected against better knowledge. An example of such a compromise would be the delivery of low-quality code to a short-term delivery deadline. The technical compromise can enable some short-term gains, TD benefit, but may cause adverse effects, so-called TD interest. TD might hamper the quality of the system (Li et al., 2015) or the software developer productivity (Besker et al., 2019). In industry, similar terms such as "workaround", "shortcut", "exposure", or "sub-optimal solution" describe this technical compromise.

In the age of production automation, various industry sectors, such as automotive, healthcare or food processing, manufacture their goods in so-called automated production systems (aPS) (Vogel-Heuser et al., 2015c). Each aPS is often a unique entity designed for a specific customer order (Birkhofer et al., 2010) and is developed on a case-by-case basis. The aPS are developed by engineers from at least three disciplines, which strongly depend on each other according to VDI/VDE 2206 (VDI, 2021) and Vogel-Heuser et al. (2015c) (cf. Figure 1). Development usually begins in the mechanical discipline in which the layout of mechanical components is defined. Based on the layout and component lists of actuators and sensors, the electrical system is designed, including the circuit diagram, the connection of actuators and sensors to the Programmable Logic Controller (PLC), and the task of distributing power. Documents from mechanical and electrical disciplines are then delivered to the software discipline, where the control software for the PLCs is developed. Because of the case-by-case development as well as the large size of the systems, the development process and life cycle of aPS include additional phases compared to typical development processes for embedded computer systems, e.g., system integration from multiple disciplines and start-up of the system at on-site customer plants (Vogel-Heuser et al., 2015b). On-site commissioning under time pressure or sudden changes of customer requirements due to different imponderables may often lead to technical compromises for short-term benefits. As such, engineers from mechanical and electrical disciplines face similar challenges (e.g., time pressure) as software developers (Besker et al., 2017). As in software engineering, TD in aPS may cause additional undesired effects. Moreover, if TD is acquired, then high costs can arise during the long lifetime of the plants – up to decades, according to (Birkhofer et al., 2010). In addition, unnoticed TD could be costly

due to the low visibility of TD. Rising economic pressure makes optimising the maintenance cost of aPS a priority.

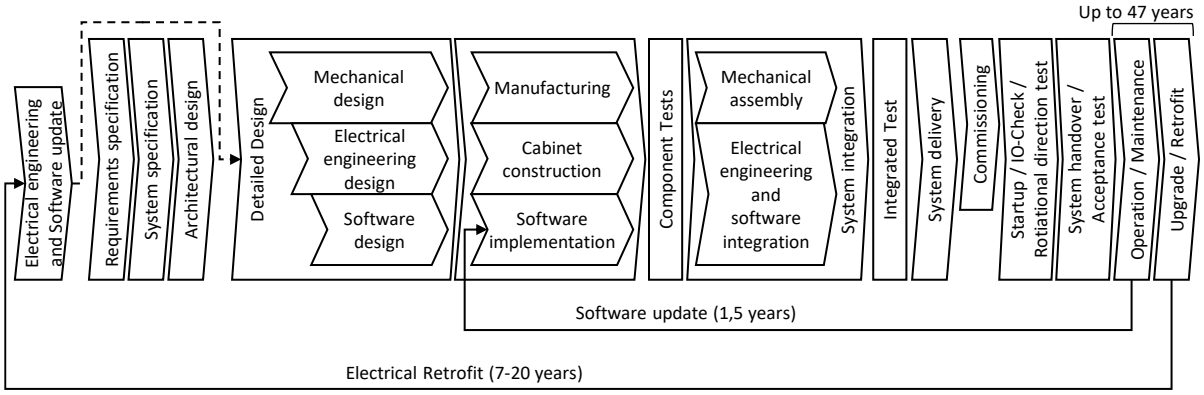


Figure 1: aPS life cycle (adapted from Vogel-Heuser et al. (2015c)).

1.1. Motivation and the scientific problem

According to Panetto et al. (2019), three aspects should be taken into account to assist management decisions in manufacturing: modelling, industrial engineering, and management. During the initial phase of the development process, modelling is an essential activity as the model is a simple knowledge-sharing method. It allows the detection of faults earlier which decreases the cost to fix the fault in later stages of the life cycle (Zimmermann et al., 2017). As "*TD can slow down engineering organisations, making it hard to manage processes where multiple domains are involved*" (Biffl et al., 2019a), TD modelling is necessary for successful project management of future systems engineering. Such modelling methodology shall take into account related tasks, phases or disciplines of the engineering process where TD occurs. Industrial experts have positively perceived graphic modelling notation in some aspects of systems engineering, such as system architecture or data analysis involving multiple disciplines (Trunzer, 2020). In addition, a system risk assessment must be considered when qualifying the plants, according to NAMUR (2022).

Since not all types of TD are harmful (TD benefit can outweigh TD interest), it is important to distinguish if TD is a problem or an investment on a specific occasion. The goal of managing TD is to find an optimal combination between a high investment return and a low-risk level. The TD metaphor has gradually expanded from coding activities to other software aspects such as design or testing, thus, providing opportunities to research and improve the TD concept. In a cross-disciplinary environment such as aPS development, "*more investigation should be conducted at companies developing automatic production systems to understand if the*

Technical Debt spans more disciplines […] *and especially if Technical Debt in one discipline (e.g. software) would generate extra costs in other disciplines (e.g. electrical engineering) and vice-versa*" (Besker et al., 2017). On the topic of systems engineering, the German National Academy of Science and Engineering (acatech) has recently mentioned: "*the focus is on model-based system design and optimisation over the entire life cycle (digital twin) and the integration of system design and project management*" (Albers et al., 2022).

Avgeriou et al. (2016a) indicated that TD always relates to cost. Besides substantial cost overruns, not "paying off" TD may hinder new feature development, e.g., existing features suffering from TD might be affected and incur unforeseen costs. However, since TD research in the aPS domain is still limited (e.g. concerning mechanical and electrical disciplines), there is lacking information such as costs or awareness of TD in aPS (***Problem1***). To manage TD, it must be identified early. The TD identification could be performed faster if a sufficient classification of TD types is available. For instance, in the classical software engineering domain, a comprehensive classification for TD in code enables automated identification with tools such as SonarQube (a widely used program for the inspection of code quality) (SonarSource SA, 2022). However, the classification of TD in aPS is still insufficient (***Problem2***). Once identified, TD might need to be presented to involved stakeholders. The representation or documentation of TD is vital since it can provide helpful information for other TD management activities, such as TD repayment or illustrating trends of TD. Thus, the amount and compounding effects of TD can be monitored. To achieve this, TD's relevant disciplines and life-cycle phases have to be illustrated visually in sufficient context (***Problem3***). Furthermore, to find an optimal combination between a high investment return and a low-risk level, TD's risk needs to be measured and assessed to support companies in prioritising, removing, and preventing TD in aPS (***Problem4***). Altogether, these result in four main problems formulated as follows.

- ***Problem1***: ***Lack of information*** about benefits, interests, sources and awareness of TD in the aPS domain.
- ***Problem2***: Insufficient classification to support TD ***identification*** in the aPS domain.
- ***Problem3***: Lack of capable methods to ***represent*** TD workflow in the aPS domain.
- ***Problem4***: Inadequate approach to evaluate TD's ***risk*** in control code documentation of aPS.

This thesis offers some selected solutions in the efforts to tackle these problems, including TD classification improvement, embedding TD perspective into modelling methods BPMN (Business Process Model and Notation) and CLD (Causal Loop Diagram), and a proposal of risk assessment method for TD in control code documentation of aPS.

1.2. Thesis structure

The remainder of this cumulative thesis is structured as follows. Chapter 2 describes the background and state of the art. Chapter 3 presents the research steps. Chapter 4 summarises the findings of the included papers. Chapter 5 discusses the research contribution, offers some conclusions drawn from the results, and describes directions for future studies. The included papers are listed in the Appendix.

2. Background and related work on Technical Debt

First, an introduction to the TD, Business Process Model and Notation (BPMN) and Causal Loop Diagram (CLD) modelling methods are given, as they are in the focus of this work. BPMN is a modelling language ideally suited to assist management decisions by integrating additional perspectives into a business process. CLD enables a graphical representation of compound issues by visualising the connection between parts of a system. Second, an overview of relevant work on TD in the software engineering domain is presented because TD was initiated in this sector and the most sophisticated techniques/strategies to handle TD originate here. Finally, related work near and in the domain of aPS is presented.

2.1. Technical Debt definition

The TD metaphor is based on the terms "debt" and "interest" from the financial domain. This metaphor was initially introduced by Ward Cunningham, focusing on software code.

> *"Although immature code may work fine and be completely acceptable to the customer, excess quantities will make a program unmasterable, leading to extreme specialisation of programmers and finally an inflexible product. Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite. Objects make the cost of this transaction tolerable. The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organisations can be brought to a stand-still under the debt load of an unconsolidated implementation, object- oriented or otherwise."* Cunningham (1993)

Since TD negatively influences the ability to produce a quality product, TD is a critical problem in software development (Li et al., 2015) (Izurieta et al., 2016). The TD metaphor provides an effective communication mechanism to describe the adverse effects of the immature code. However, the community has realised that TD is also a software development artefact primarily incurred unintentionally and detected during later phases of the software development process (Izurieta et al., 2016) (Avgeriou et al., 2016b). Thus, a more recent definition was given:

> *"In software-intensive systems, technical debt is a collection of design or implementation constructs that are expedient in the short term, but set up a technical*

> *context that can make future changes more costly or impossible. Technical debt presents an actual or contingent liability whose impact is limited to internal system qualities, primarily maintainability and evolvability."* Avgeriou et al. (2016b)

The future changes are more costly for software suffering from TD, as TD could force the engineers to perform undesired time-consuming activities to progress the development work and deliver high-quality software. This wasted time may negatively affect efficiency and reduce productivity of the software developer (Besker et al., 2019).

2.2. Business Process Model and Notation

In their work related to organizational learning, Argyris and Schön (1978) indicated that models, i.e. conceptual maps, could be used for formulating, implementing, and reviewing a strategy. The models allow practitioners to simplify complex real-world data to support continuous learning and analysing of different situations for making decisions.

Specified as an ISO standard, BPMN (OMG, 2011) is often employed for illustrating the business process. BPMN defines some basic shapes (cf. Figure 2), such as activity (work or task to be performed), event, gateway (to control the workflow), flow and lanes (to organize the activities). A fictitious example of a BPMN diagram showing the activities conducted by different disciplines of a supplier is illustrated in Figure 2.

BPMN is extensible and allows the models to be enhanced with additional concepts. The original BPMN elements can be adapted to illustrate characteristics of a particular domain (Stroppi et al., 2011) (Zarour et al. (2019). For instance, additional symbols or icons can be added to the activity to represent domain-specific information associated with a task.

Many BPMN extensions have been developed to tailor it to the needs of various domains, according to Zarour et al. (2019). For instance, Braun et al. (2016) proposed a BPMN extension, namely BPMN4CP, to enable multi-perspective modelling of clinical pathways for the healthcare domain. Chergui and Benslimane (2018) focused on business process models in security. In particular, they developed a BPMN extension that allows practitioners to specify and enforce compliance and security requirements for business process-driven enterprise systems. In the production domains, Bocciarelli et al. (2017) proposed an extension for modelling aPS in the context of Industry 4.0. The extension allows representing the cyber-physical systems as resources supporting business process execution. Graja et al. (2016)

proposed BPMN4CPS for modelling cyber-physical systems. The extension enables designers to explicitly model concepts of cyber-physical systems such as actuators or sensors. The extension in Yousfi et al. (2015) aims to allow the use of ubiquitous computing (e.g. sensors and smart readers) for business process improvement. Abouzid and Saidi (2019) proposed an extension in which the timing of each operation in the manufacturing process is considered. The extension in Polderdijk et al. (2018) allows human physical risks, such as heavy lifting, to be modelled in manufacturing processes.

The BPMN is widely adopted in academia and industry due to its benefits, such as standardization or tool support; thus, the costly development of a domain-specific modelling language from scratch is avoided (Braun and Esswein, 2014) (Braun et al., 2016). For instance, business processes described in BPMN can be executed by various engines such as Activiti, jBPM or Camunda BPM (executability). This modelling technique can be employed to integrate further perspectives to improve the decision-making process in the aPS domain (Vogel-Heuser et al., 2022a). Hence, an integration of implicit knowledge such as TD into aPS development and maintenance process seems promising with BPMN.

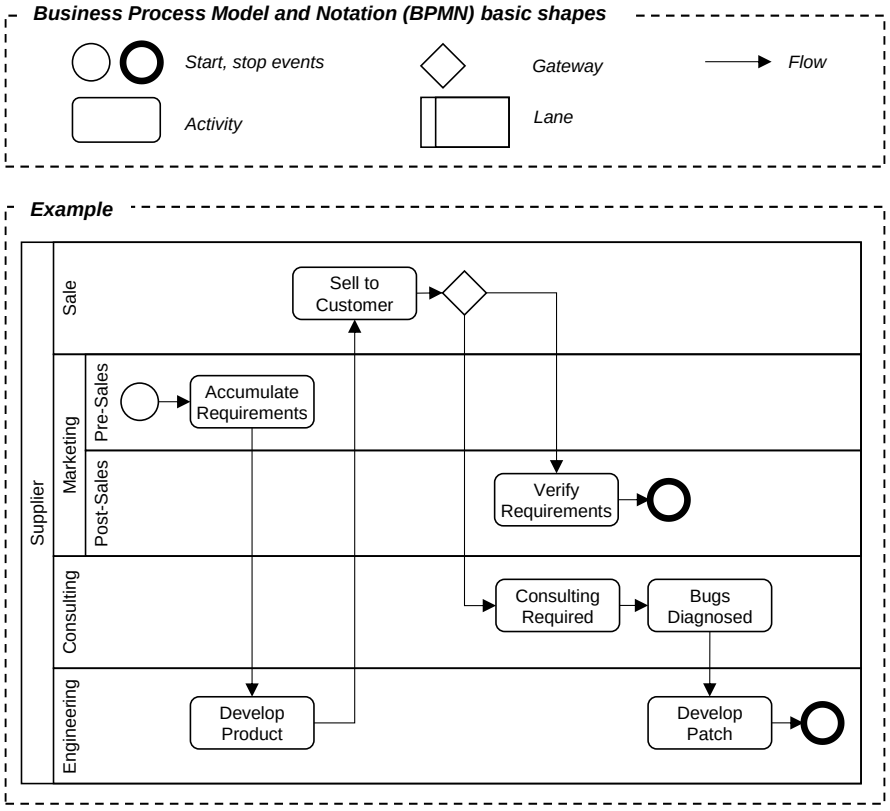


Figure 2: An example of Business Process Model and Notation (adapted from OMG (2011)).

2.3. Causal Loop Diagram

Under the theme of management science research, System Dynamics modelling was coined by Forrester (1961) as an application of Systems Theory. The System Dynamics approach aims to establish a basis for resolving management problems. Mainly, the approach focuses on explaining the problematic behaviours and how they appear with the "feedback" concept.

CLD, a method introduced in the System Dynamics approach, assists in modelling the interior structures of a system, i.e. causal relations between the variables. CLD notation is exemplarily illustrated in Figure 3. The variables are connected by causal links, denoted by arrows representing the influences between the variables. In the example in Figure 3, the Population variable is influenced by two variables, Births and Deaths. Each causal link is assigned a polarity, either positive (+) or negative (-), to indicate how the dependent variable changes when the independent variable changes. The positive link represents a situation where the variable at the head of the arrow changes in the same direction as the variable at the arrow tail (both increasing or decreasing). For instance, an increase in the Births variable will lead to an increase in the Population variable; thus, the Link Polarity (+) is assigned (cf. Figure 3). On the contrary, the negative link describes the case where two variables change in opposite directions.

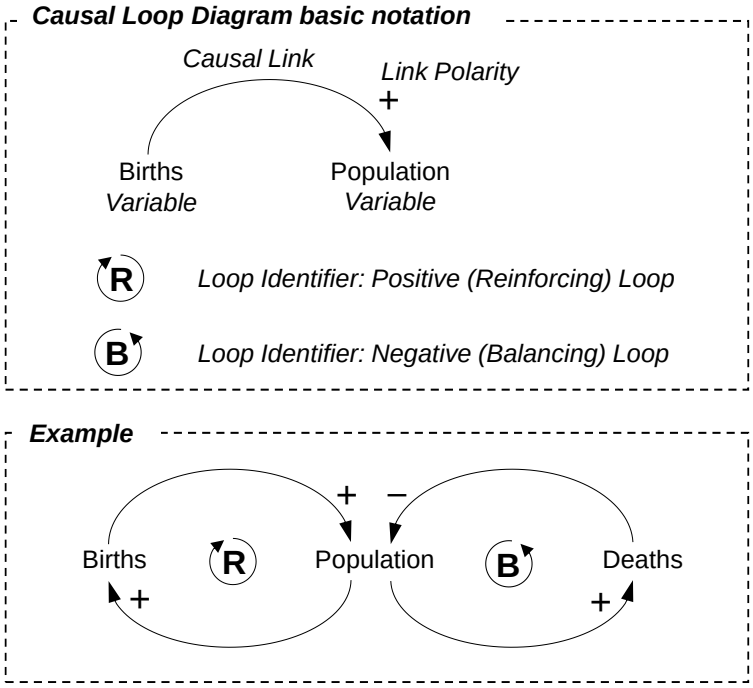


Figure 3: An example of Causal Loop Diagram (adapted from Haraldsson (2004) and Sterman (2000)).

According to Sterman (2000), system behaviour is often influenced by two feedback loops, namely positive (reinforcing) and negative (balancing), which are denoted by loop identifiers (cf. Figure 3). One can begin with a hypothesis, e.g., assuming an increment in a variable, and follow the circle to determine the loop type. A loop is denoted as reinforcing if the value of the initial variable ends up with the same result as the initial hypothesis after travelling through the loop. In the left loop of the example in Figure 3, if the population increases, it will cause a rise in births, leading to an increase in the population. Thus, the left loop is marked as reinforcing. In contrast, the value of the initial variable in a balancing loop will change in opposite directions after the loop. In other words, while the reinforcing loops amplify a behaviour, the balancing loops stabilise the behaviour. Therefore, the CLD enables practitioners to conceptualise and construct circular connections and feedback on a problem based on identifying feedback loops in the variables (Haraldsson, 2004).

The System Dynamics modelling approach can assist practitioners in predicting the system's overall behaviour and in what sequence different causal links will behave (Haraldsson, 2004). For instance, to anticipate the trend of a variable, one can perform loop dominance analysis on a CLD model to derive Reference Behaviour Patterns (RBP), i.e. a graphical representation of the change over time of a variable, can be analysed in the CLD (Sterman, 2000). A loop dominance describes which part of the feedback in a CLD is the strongest (most active) at a given time. Thus, possible dynamic behaviours of the variables can also be examined.

Furthermore, the System Dynamics approach can allow knowledge transfer between disciplines (Haraldsson, 2004) or organizational silos (Rehan et al., 2014). For instance, Beck (2003) applied CLD in testing to analyse and prepare for different types of changes in development (test-driven development). From the management domain, the System Dynamics approach has gradually been applied to other domains. For example, Davies (2018) employed CLD to develop the Market Dynamics Model in the business and economic domain. The Market Dynamics Model was established from interview data aimed at studying customer behaviour, e.g. underlying causal drivers when the customers decide to switch services. Therefore, this modelling approach can be transferred or adapted to study problems in other fields, such as TD in software engineering (Franco, 2020).

2.4. Technical Debt in the classical software engineering domain

TD has received significant attention from researchers in the classical software engineering domain. Avgeriou et al. (2016b) stressed the importance of TD research for product quality. Starting with coding, the TD metaphor has gradually been applied to other software engineering aspects, such as design, documentation, requirements, and testing (Li et al., 2016). In agreement with the TD (sub-)types explored by Tom et al. (2013), a TD classification based on the software development process phases was presented by Li et al. (2015). The Li et al. classification tree categorises TD into different types according to the phases of the software development life cycle, and each TD type is further classified into sub-types which use the causes as criterion. The tree includes ten types of TD: requirements TD, architectural TD, design TD, code TD, test TD, build TD, documentation TD, infrastructure TD, versioning TD and defect TD. In addition, definitions for those TD types are provided. For example, "*Documentation TD refers to insufficient, incomplete, or outdated documentation in any aspect of software development. Examples include out-of-date architecture documentation and lack of code comments*" (Li et al., 2015).

Due to the long-term negative effect and its low visibility, TD could become a severe problem if ignored (Martini et al., 2015). Failure to monitor TD has resulted in unexpectedly significant cost overruns in many software development projects (Guo et al., 2011). Besides substantial cost overruns, Seaman et al. (2012) presented other examples of the penalties for not "paying off" TD, such as quality issues, the inability to add new features without disrupting existing ones, and the premature loss of a system. Previous studies (Fairley et al., 2017) (Ramasubbu and Kemerer, 2014) (Holvitie and Leppanen, 2015) showed how TD influences a project's planned, reported and actual product delivery date. TD could also impact the organisation's profitability, according to Kruchten et al. (2012), Ampatzoglou et al. (2015), Snipes et al. (2012), and Martini and Bosch (2016). Even if software engineers requested investments to remove TD, executives might not approve after inquiring about their business value. The individuals choosing to incur TD (e.g. designers or developers) could often be different from those responsible for servicing the debt (e.g. maintenance staff). In addition, decisions regarding TD still have only rarely been quantified, and organisational gaps among the business, operational, and technical stakeholders could incur more debts. Most information regarding previous decisions is available as "tribal memory," which is an unreliable source of historical data due to forgetfulness and the change in team members over time. TD decisions

have been managed in an ad hoc manner, according to Klinger et al. (2011), Holvitie et al. (2016), and Izurieta et al. (2017).

Experts from both academia and industry have continuously expressed the importance of ensuring that those who design, develop, and maintain software understand TD's potential consequences. These consequences are not only economic ones but may also include, e.g., the motivation of employees (Szabados and Kovac, 2015) (Avgeriou et al., 2016b) (Stopford et al., 2017). Monitoring and managing TD in the architecture could enable analysis earlier in the development cycle and keep the project on track. In addition, the findings of Lim et al. (2012) recommend increased communication about TD and making TD more visible to all involved stakeholders. The TD paradigm has been increasingly recognised as an essential means for technical communication (Martini et al., 2022). Similar to financial debt, TD might sometimes be necessary, e.g. to meet the project schedule (Brown et al., 2010) (Wiklund et al., 2012). Future interest payments can be reduced by restructuring of architecture or code refactorisation, i.e. TD recovery (Brown et al., 2010) (Martini and Bosch, 2016).

Codabux and Williams (2013) studied how software development teams manage TD. The publication did not cover a specific TD item (i.e., specific TD in a system) but instead focused on the overall process of managing TD and understanding what can be considered as TD interest. The research suggested that 20% of the development time should be collectively invested for TD reduction. However, there was a lack of consensus among the participants in the study related to TD terminology and thus, showed demand for developing a scheme to characterise various types of debt. The work by Nugroho et al. (2011) proposed an approach to rate the quality of a software system. The levels of quality in the rating system were based on the findings of previous statistical studies and ISO/IEC 9126 sub-characteristics of maintainability. The interest of TD would be the extra cost spent on maintaining software due to poor quality, such as unnecessarily complex code. The work also proposed some formulas to calculate the increment of TD over time; however, other development artefacts, such as documentation, were not yet considered.

The work by Guo et al. (2011) aimed to track a single delayed maintenance task in an actual software project throughout the project life cycle. They focused on TD in the maintenance phase. The paper presented how postponing paying TD can result in even higher interest. However, the scope of the work was limited to a pure software application (Microsoft Exchange program). Based on Lehman's laws, Franco (2020) employed CLD to model

relationships of software maintenance factors. The work suggested that different software maintenance strategies can influence the amount and interest of TD. Thus, modelling TD may support understanding and analysing interdependencies such as TD causes and effects. Although the benefit of TD was considered, no TD benefit factor was included in the presented model. In addition, modelling TD artefacts in a specific use case were not addressed in the proposed approach, as the research focus was on the strategic level.

Zazworka et al. (2011) addressed the question of which debt should be paid first in case several cases of TD are detected. The authors investigated the design TD caused by so-called God Classes that control many other objects in the system. The study used the metrics of the change effort and defect likelihood to rank the various cases of TD. Regarding the effort, the authors reasoned that a God Class has low refactoring effort if it is only slightly outside the thresholds. Regarding the impact, the authors argued that classes with significant change and defect likelihood are likely to impact the quality significantly. The impact on quality was defined as the interest of the debt, which could be determined by some software quality characteristics such as correctness or maintainability (Zazworka et al., 2014). Digkas et al. (2021) conducted a case study to measure the risk of extra maintenance costs incurred by the existence of TD items in a software system (TD interest). They indicated that if the interest of TD is small, the effect of TD is negligible; if the interest is significant, the system becomes unsustainable. The study's results suggested that the risk metrics, e.g. probability of TD interest, can be used to prioritise TD. However, the study did not yet consider particular characteristics of software artefacts and development practices. Thus, further research is necessary concerning the risk and prioritisation of TD, particularly regarding other development artefacts such as documentation.

The topic of TD has attracted significant attention (cf. Figure 4), making it an important research topic. However, the research on TD management activities such as representation (modelling) or risk prioritisation is still poorly addressed in the current literature (Rios et al., 2018) (Lenarduzzi et al., 2021) (Melo et al., 2022). In the case of TD that relates to non-code artefacts, TD should be represented in an appropriate form and documented for further management. Unfortunately, the existing approaches only focused on a subset of activities such as identification, measurement, or repayment of TD (Li et al., 2015). It is explained that in the classical software engineering domain, code analysis tools (e.g. SonarQube) can be used to browse TD and the results can subsequently be exposed to stakeholders (Li et al., 2015). Nevertheless, some research results from the classical software engineering domain can be

transferred to the aPS domain, e.g. the CLD approach from Franco et al. (2020). Still, these existing methods may not fully meet the challenges of interdisciplinary development (i.e. mechanics, electrics and software); thus, those methods would require adaption for the aPS domain.

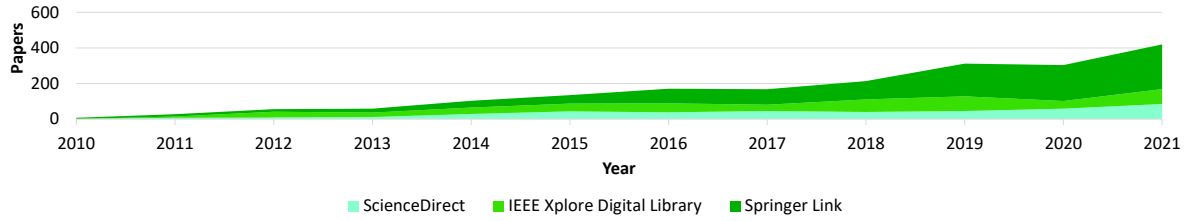


Figure 4: Increment of research interest towards TD in the software engineering domain in recent years (number of publications from the search for the keyword "Technical Debt" in three well-known sources for scientific research).

2.5. Technical Debt near the automated production systems domain

Martini et al. (2015) investigated the causes of TD accumulation in embedded systems. The study focused on the architecture of the software. They reported the danger of TD accumulation that eventually, in some cases, made future development impossible due to the effects of the chosen sub-optimal solutions in software architecture. According to Li et al. (2015), these sub-optimal solutions were classified as architectural TD. Different causes of architectural TD were identified, including uncertainty of use cases in early stages, uncertainty of impact, parallel development, time pressure, new business requirements, priority of features over product, and uncompleted refactoring. Based on the Grounded Theory from Strauss and Corbin (1997), Martini et al. (2015) developed a crisis point model and checked it through interviews. The case study presented a more detailed model, including several variables of an accumulated architectural TD, related phases and refactoring activities. Based on these initial models, different refactoring strategies were considered to infer trends of TD. However, the proposed models were only partially confirmed. Martini and Bosch (2016) reported a use case where costly refactoring or a redesign would be necessary to add new features to the system if the TD accumulation is mismanaged.

Also focusing on the architectural aspect of the software, Martini and Bosch (2017) presented a model of the causes for architectural TD accumulation, the classes of architectural TD, and the phenomena caused by debt. They reported an architectural TD use case related to uncompleted refactoring for code duplication. The software architects identified a violation in

which three different methods were used to communicate among components. This violation created difficulties for developers during development since there was confusion in choosing the correct communication method. The architects planned to replace the three existing protocols with a new one to keep the system consistent. However, some unintended consequences related to the implementation of the to-be-removed protocols were found during the refactoring processes. Unfortunately, such effects were not taken into account during the estimated refactoring time. Thus, the developers were unable to eliminate the existing protocols because of the time constraints to finish the refactoring, which led to the inclusion of a fourth protocol in the system. Moreover, the management could not prioritise such refactoring again, leaving the original issue unresolved. This accumulation of TD due to uncompleted refactoring will require a more significant refactoring effort to resolve the reported TD completely.

Vogel-Heuser and Bi (2021) investigated the interdisciplinary effects of TD in companies producing mechatronic products. Significant TD types and sub-types were collected using the interview method. The work identified a significant amount of TD taken in the early stages of the life cycle and reported that many unresolved TD are posing a potential risk to the systems. Based on the results in Vogel-Heuser and Bi (2021), Bi et al. (2021, 2022) conducted a follow-up investigation on the frequency, impact, and patterns of the collected TD (sub-)types. The analysis reported an accumulation of TD at the beginning of the life cycle. Due to the contagiousness of TD effects in the life cycle, disciplines in later life-cycle phases are more affected by TD.

2.6. Technical Debt in the automated production systems domain

This part highlights the characteristics of aPS, followed by requirements for TD research in aPS, and concludes with a description of the related work in the aPS domain.

2.6.1. Characteristics of aPS

aPS are a subset of mechatronic products. The development process of aPS is more complex than the classical software engineering process since aPS are long-living, large and complicated systems. aPS development involves collaboration across multiple disciplines such as mechanical, electrical and software engineering (cf. Figure 1). In the following, aPS development characteristics are highlighted at different scopes from software to boundary conditions.

Software

aPS software often meets various challenges which are usually not considered by software analysis techniques established in high-level language programming or embedded systems software (Lüder et al., 2005). Some distinct characteristics in developing aPS software regarding programming languages, architecture, design concepts, and functions are given as follows.

There are differences in programming languages for aPS software running on PLC compared to the languages used in the classical software engineering domain. The aPS software is often developed following IEC 61131-3 standard (IEC, 2013). IEC 61131-3 specifies three types of Program Organization Unit (POU): (1) Function (FC), (2) Function Block (FB), and (3) Program (PRG). Each POU is often comprised of a comment header, a variable declaration part and an implementation part. The IEC 61131-3 compliant languages consist of three graphical languages (i.e. Ladder Diagram, Function Block Diagram and Sequential Function Chart) and two textual languages (i.e. Structured Text and Instruction List). Although the object-oriented extension has been incorporated into the IEC 61131-3 standard for a while, it is still barely adopted in industrial practice (Neumann et al., 2020).

Furthermore, compared to classical embedded systems, the *architecture* of PLC controlling aPS has some particularities. Instead of using an event-driven real-time operating system, PLC follows a cyclic execution model with monitored cycle times (Ulewicz, 2018). aPS application engineers or technicians sometimes lack the necessary software backgrounds because they are mostly educated to control the technical process. These particularities pose significant challenges to the development of aPS software (i.e. control software), which strongly differs from classical embedded systems applications (Wilch et al., 2022). For example, control software needs to enable software changes while the aPS are operating and manual mode to recover from faults by operator intervention (Vogel-Heuser et al., 2015c).

In control software, the modularity, software hierarchy levels or data exchange between modules often follows different design concepts than classical software engineering (Neumann et al., 2020). Particularly, the control software architecture often follows a monolithic structure, which hinders maintainability (Fuchs et al., 2014). Moreover, while design patterns are often used in high-level programming languages to resolve common and recurring problems, reusable solutions offered by design patterns are still limited in IEC 61131-3 programming (Fuchs et al., 2014) (Cruz et al., 2019). In some companies, control software is modularized

based on the physical layout of the machine (Neumann et al., 2020). The architecture has to cope with particular requirements, such as the strict real-time of the tasks running in one PLC cycle, e.g. to provide stability in the manufacturing process. Another characteristic is that global variables are widely used to exchange data or for fault handling. Industrial practice shows that control software depending on global variables has relatively low modularity (Vogel-Heuser et al., 2017a).

Compared to classical software, functions realised by the control software have divergent characteristics or importance (Wilch et al., 2022). Besides application logic tasks, control software has to realise extra functional tasks such as hardware control, operation mode switches, or fault handling (Neumann et al., 2020). While human behaviour is the predominant factor interacting with the classical software, hardware behaviour is the main factor driving the control software of aPS (Ulewicz, 2018). Thus, control software design must adapt highly to the hardware behaviour because many faults are connected to hardware failures. Furthermore, some additional hardware constraints, e.g. usage of hardware with lower quality than specified to reduce costs (Vyatkin, 2013), must be considered in aPS development.

Due to the above differences, traditional software engineering methods from computer science must be adapted to be applied to control software development (Neumann et al., 2020). In addition, due to Industry 4.0 requirements, such as adaptability to a wide variety of products, the complexity of control software is increasing (Fay et al., 2015) (Seitz et al., 2021). The growing complexity is also due to the need to improve the degree of automation in production (Fay et al., 2015) or to establish more distributed production facilities (Neumann et al., 2020). Therefore, there is a growing demand for methods supporting automation engineers during the development process.

Boundary conditions

Several boundary conditions are commonly encountered in aPS development, including interwoven disciplines, automated process types, company-specific constraints, system requirements and tiers (a popular term in the automotive industry describing companies along the supply chain).

First, there is a strong coupling between the involved disciplines during aPS development. For example, exchanging or replacing mechanical or electrical equipment often induces a software change, either in the form of an adaption of existing POUs or the implementation of new POUs

(Vogel-Heuser et al., 2015b). Due to these constraints involving multiple disciplines and the increasing size or complexity of software systems, developing aPS is a complex task.

Second, the type of automated process is another factor influencing the development of aPS. Specific requirements need to be fulfilled for different types of manufacturing processes. For instance, continuous processes may have sequential time constraints, creating strong dependencies on the components and hampering the system's modularity. Additionally, to keep competitive, each aPS customer might have their own specific or tailored manufacturing process, i.e. process-specific constraints, which must be considered during aPS development.

Third, the development of aPS is often influenced by other company-specific constraints, such as the company's location(s) or the engineers' backgrounds (Neumann et al., 2022). For companies developing aPS at multiple locations, good modularization and documentation would be a prerequisite for, e.g., reusing or adapting the components to the company-specific requirements. Different disciplines developing the aPS might have different perspectives regarding the quality of aPS. In addition, the aPS quality might also be influenced by engineers with different backgrounds, e.g. in-house module developers or on-site technicians (Vogel-Heuser et al., 2022c).

Fourth, aPS needs to meet requirements in systems engineering such as dependability (i.e. availability, reliability, security, integrity, accessibility, and maintainability). Depending on the industry, specific standards can exist (e.g., OMAC for packaging machines) or legal requirements (e.g., in the medical sector) are needed to meet those quality requirements. The aPS development is often validated with GAMP (Good Automated Manufacturing Practice) (ISPE, 2022) (NAMUR, 2022), which is a risk-based approach accepted in the pharmaceutical and food industries to regulate manufacturing systems. Risk priority in the GAMP method is based on three factors severity, probability, and detectability using a multiplicative approach: factors are multiplied by each other in two stages. First, the severity is offset against the probability. This calculation determines the risk class. Second, the risk class is then offset against the detectability and results in a risk priority.

Fifth, the aPS development often involves multiple companies at different tiers. Tier 1 (customer) includes companies producing the consumers' final products, such as car wash machines. Tier 2 consists of the machine (MM) and plant manufacturers (PM) representing companies making the machines or building the plants for Tier 1. Tier 3 contains companies producing the devices or platforms, such as sensors for Tier 2. Tier 3 can also provide devices

to Tier 1. While the MM can integrate and verify the systems before delivery, the PM is often not able to test the whole system due to the heaviness and large dimensions of the modules, which sometimes equal small plants. Thus, the customer site is often the place where the first full integration test for aPS is conducted (during the commissioning phase). Unfortunately, this period is usually short because the customer wants to start production as soon as possible; therefore, engineers are often under high pressure, a common trigger for TD (Vogel-Heuser et al., 2015b). Hence, the maintenance cost for sub-optimal solutions can be high since aPS have a long life-cycle time.

2.6.2. Requirements for TD research in aPS

As "[life-cycle] *cost reduction remains the main driving force in automation*" (Vyatkin, 2013), research on TD in aPS is vital. Rabiser and Zoitl (2021) indicated that adopting promising software engineering techniques and technologies is necessary to address particular challenges in production automation. Specifically, methods and tools have to support engineering across disciplines and system boundaries, according to Brings et al. (2019). Well established in computer science, the model-based engineering approach, e.g., graphical modelling, represents a potential means for efficient cross-disciplinary engineering, variability management, or coping with the high software complexity in technical systems (Tichy et al., 2008) (Zimmermann et al., 2017). In addition, the approach should take the systems' life-cycle phases into account (Vogel-Heuser et al., 2015c). Unfortunately, model-based software engineering is still barely used in industrial automation (software) development, according to Vogel-Heuser et al. (2022b).

While some code TD, like code with poor quality or code that violate coding practices, can be automatically identified and modelled using static code analysis methods in software engineering, the identification of TD related to mechanical and electrical artefacts is not trivial due to other languages (graphical ones) and insufficient classification (***Req-Classification***). It is assumed that an improved TD classification can assist in identifying TD more effectively. Moreover, once identified, the TD is sometimes not sufficiently documented due to a lack of methods to model TD in aPS. Therefore, the mechanical or electrical TD analysis to monitor the effects, amount, and interest of TD is hampered. Since graphic materials are more effective than text in human understanding, TD modelling methods with graphical notation would help assist the presentation of TD to involved stakeholders (***Req-Modelling***). Graphical modelling is assumed to be applicable to represent all types of TD. So far, the work on risk assessment of TD in the aPS domain is limited (***Req-Risk***). It is assumed that the risk associated with a

specific TD can be measured by considering the related aspects driving that TD. Last but not least, studies with data collection from industrial companies are necessary to gain more empirical knowledge of TD phenomenon in the aPS domain (***Req-Survey***). Under the assumption that results from an online survey are reliable, the findings obtained from case studies can be validated. Hence, the formulated requirements listed in Table 1 need to be fulfilled to assist in analysing, monitoring and reducing TD in aPS.

Table 1: Rating scheme of requirements to evaluate relevant work (related problems are listed in Introduction section).

Req-Classification **– Granularity of TD classification** Related problem: ***Problem2*** *(identification*)	
+	Fine-grained classification (considered TD types and sub-types), multiple disciplines considered, and examining life-cycle phases of aPS.
○	Limited classification (coarse-grained, no multiple disciplines considered, or not examining life-cycle phases).
-	No classification.
Req-Modelling **– Support of modelling TD use cases with graphical notation** Related problem: ***Problem3*** *(representation)*	
+	Support of graphical modelling of TD use cases considering: (1) discipline and life-cycle-phase aspects to illustrate the propagation of TD in aPS, (2) abstraction, interpretation and executability.
○	Limited support of graphical modelling of TD use cases (not considering discipline or life-cycle-phase aspects of aPS).
-	No consideration of graphical modelling.
Req-Risk **– Support of assessment of the risks associated with TD** Related problem: ***Problem4*** *(risk indicator*)	
+	Approach considering: (1) assessment of the risk aspect associated with a specific type of TD, and (2) extensibility, flexibility and feasibility to automate.
○	Approach only consider risk aspect of TD in general, not a specific TD type.
-	No consideration of assessment of risk aspect of TD.
Req-Survey **– Industrial survey** Related problem: all identified problems from ***Problem1*** *(lack of information)* to ***Problem4*** *(risk indicator*)	
+	Including (1) quantitative and qualitative industrial surveys, and (2) companies at different tiers working with aPS.
○	Limited consideration of data collection (e.g. data mainly collected from demonstrators or personal experiences or from one company).
-	No data collection.

2.6.3. Related work in the aPS domain

Vogel-Heuser et al. (2013) showed that the evolution of automation software for machines and plants tightly correlates with the evolution of its mechanics (including mechanical, hydraulic and pneumatic disciplines), electrical/electronics, and software components. Vogel-Heuser et al. (2017a) studied typical architectures to identify the state of the practice of reusability and modularity in industrial aPS companies. It was found that these practices supported a reduction of the development effort for aPS. However, it is unclear how different reusability and modularity levels affect TD, as the work did not consider the TD aspect.

Vogel-Heuser et al. (2018) presented a typical workflow of control software library module development. The workflow starts with module design by module developers. Next, the module application is implemented by application engineers. Finally, start-up technicians integrate and test the modules on-site, i.e. in the real production environment. If the source code of an existing library is complex, application engineers and start-up technicians often quickly create a new module instead of improving the existing library module. From a TD perspective, this process can be considered taking a TD since more development effort will be required to maintain both old and new modules and merge them, i.e., paying the interest of TD. The work proposed some modularity metrics to evaluate the maturity level of a module library; however, the TD concept is not yet considered in this work.

Vogel-Heuser et al. (2020a, 2022a) proposed an extension of BPMN, namely BPMN+I, to support decision-making in innovation management for aPS, where the aspects of multiple disciplines and life-cycle phases are considered. The extension introduces eight new notational elements and two sub-diagrams to illustrate the cooperation between disciplines. The presented use case describes a procedure derived from surveys in plant manufacturing focusing on the start-up phase and on-site decision support. The procedure includes two alternative solutions for a short-term management decision in the start-up phase of aPS. However, the TD concept was not introduced in the publication.

Due to a closely interwoven development and maintenance process between disciplines, TD in aPS is composed of decisions from multiple disciplines rather than only those made in the software discipline. aPS have three dimensions: mechanics, electrics, and software. Moreover, compared to software engineering, aPS engineering includes additional phases, like control cabinet design and manufacturing. Also, maintenance activities include evolving the designs and managing various variants and versions (Vogel-Heuser et al., 2015c). Thus, existing

approaches to cope with TD from the software engineering domain do not cover all aspects of aPS development and maintenance. To gather enough information on TD spanning across the disciplines and/or across the life-cycle phases of aPS, discussions with personnel and examinations on engineering artefacts from multiple disciplines along the life-cycle phases are required.

Vogel-Heuser and Rösch (2015a) and Vogel-Heuser et al. (2015b) presented some examples of TD in aPS and thus made a first step to applying the TD concept to the aPS domain. For example, the melted isolation of wiring of a motor was poorly repaired by a local service who, as was unfortunately often the case, lacked experience and did not have the required equipment, such as fixtures, for this complex motor. In this example, an appropriate spare motor was not in stock and a low-quality repair or replacement was a necessary option to continue the production operation. However, those temporary solutions were often forgotten and not updated in the documentation when the system went back into operation. As such TD was not resolved, the interest of TD could be a less accurate motor behaviour, overheating and, in the worst case, fire due to the overheating resulting from the continuous operation with poor quality material. An initial enlargement of TD classification for aPS was presented (cf. Figure 5). However, the presented TD items were limited, lacked systematic large-scale analysis, and were collected from the heuristic use cases.

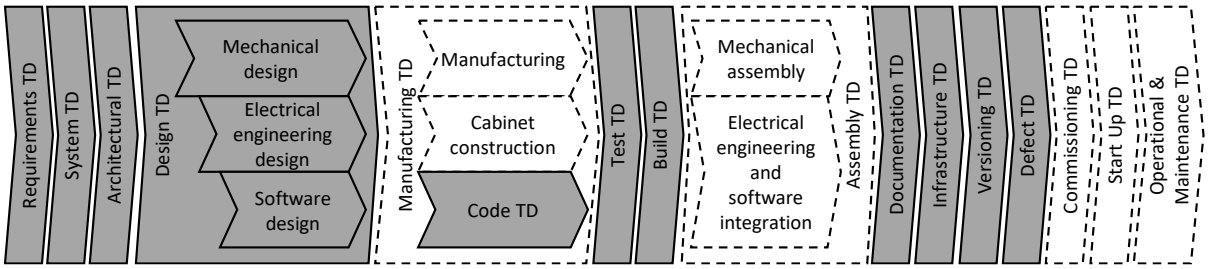


Figure 5: Enlarged TD types for aPS (grey boxes: TD types introduced in Li et al. (2015); dashed-line boxes: complemented items) (adapted from Vogel-Heuser and Rösch (2015a)).

Some TD items in aPS were identified in case studies (Vogel-Heuser and Neumann, 2017c) (Capitán and Vogel-Heuser, 2017). A few TD aspects of classical software engineering were extended to control software development. Vogel-Heuser and Neumann (2017c) focused on concrete TD types in fault handling, modes of operation and safety aspects in control software. Capitán and Vogel-Heuser (2017) adapted some software metrics from high-level programming languages in the software engineering domain to indicate TD in control software. Bougouffa et al. (2018) indicated some TD in PLC code using a static code analysis tool.

However, the above studies were performed on a laboratory plant and addressed only selected aspects of the control software.

Vogel-Heuser et al. (2017b) introduced KAMP4aPS as a change effort estimation method working on cross-disciplinary metamodels. Through the use of the model, one can simulate modifications from a change request. A fine-grained task list can be generated based on change propagation in the model. These fine-grained tasks enabled the ease of estimating the required change efforts. Based on KAMP4aPS, Cha et al. (2018) embedded the TD concept into the decision-making workflow considering multiple disciplines to assist in the solution selection process for the aPS domain. The TD interest is evaluated in an exemplary multidisciplinary change scenario. Although the KAMP4aPS approach seems to allow automated analysis, the extensibility and flexibility aspects were not presented, and the two above studies were performed on a laboratory plant only.

Besides intentional TD, which was explicitly taken and tracked, there is unintentional TD, i.e. TD taken unknowingly (Li et al., 2015). A change which leads to unknown inconsistencies can be perceived as an unintentional TD. Feldmann et al. (2014) proposed an ontology-based approach to detect inconsistencies between requirements and test cases. With this ontology-based approach, unformatted data (e.g. unstructured text) is transformed into a formal structure which enables processing, querying and detecting the inconsistencies. However, the TD concept was not introduced in this research yet. Ocker et al. (2019) proposed a method to identify and assess the criticality of TD based on conformance violations and interdisciplinary inconsistencies checks. Although the proposed approach seems to be extensible, flexible and allows to be automated, it requires a set of formal knowledge bases, which is challenging to establish due to the diversity of viewpoints related to the aPS development (e.g. disciplines, products, processes, or resources). In particular, the various metamodels of multiple disciplines are required, ranging from automation to psychology to finances, covering the whole life cycle and considering different stakeholders (Kohn et al., 2013). For metamodels in automation, there is not only one reasonable metamodel, but various metamodels may also appear depending on the purpose of the models, such as different perspectives on the systems (Cha et al., 2020). In addition, Ocker et al. (2019) only focused on the design TD of a laboratory demonstrator.

In Bosch, Tichy & Martini's working group, Besker et al. (2017) focused on the cost of TD interest and the awareness of TD. They reported that while the price for TD interest was 32%

of the aPS' development time, there was only moderate awareness of TD. However, the research was only conducted with the software discipline of one aPS company in the Scandinavian region.

In Weippl's working group, Brenner et al. (2019) expressed that TD (e.g. insufficient requirements) might not only induce defective software but could also result in defects of physical parts (e.g. aPS). Three models of causes and effects from TD were presented. They suggested that to monitor the rise of TD, continuous risk analysis is necessary. However, the focus was on software security, and only fictive examples were examined.

In Biffl's working group, Biffl et al. (2019a) employed the Quality Function Deployment method (tabular approach) to assess the risks of TD related to process documentation and configuration management. The focus was on engineering data exchange, e.g. requirements supporting software engineers. Although the approach seems to be extensible, the preliminary results from the study (i.e., a list of TD effects, items, causes, and their relationships) require further validation since the data was collected at only one company. They noted that "*engineering process description may highly depend on the context, domain, and organisation, thus future case studies should consider these variation points*." The feasibility of being automated and flexibility of the approach were not presented in the publication. In addition, graphical modelling was not considered in the study yet. "*The relations between TD effects, items, and causes highlighted the need for better representations for production knowledge as inadequate context and artifact descriptions lead to high efforts, in particular, for software engineers, and might result in economic project failure*", Biffl et al. (2019a) noted.

Biffl et al. (2019b) studied the data exchange process and reported some TD items related to engineering data models and instances. The study indicated that the identified TD items caused "*adverse effects on project effort, cost and duration as well as data quality*." A preliminary TD cause-and-effect model was reported, and the model was found helpful for discussing similarities and differences in other companies. Biffl et al. (2019c) reported three TD items related to engineering description languages. However, these two studies were only conducted with one company. In addition, risk measurement or prioritisation was not considered yet.

Waltersdorfer et al. (2020) described some TD items in the data exchange process between four data providers and two data consumers. The data providers include mechanical, electrical, fluidics, and software engineers. The data consumers are comprised of project managers and

simulation engineers. The study focused on the development phase; thus, TD at subsequent phases, such as commissioning or maintenance, was not in consideration.

Based on the systematic classification of Li et al. (2015), a synthesis across previous work is conducted to present the state-of-the-art TD classification for aPS. So far, the reported TD in previous studies (cf. Figure 6) are still mainly collected from some phases of the software discipline of aPS (e.g., architecture, code or documentation); thus, further study is necessary to explore TD in other phases and disciplines of aPS.

In conclusion, despite recent progress by the research community in the classical software engineering domain to understand TD, TD research in the aPS domain is still limited. The relevant work on TD is summarised in Table 2 and shows that none of the evaluated studies succeeds in fulfilling all the identified requirements. It is primarily due to the specific characteristics and boundary conditions of aPS, the current research results from computer science could not be (directly) applicable. In addition, different disciplines and tiers working aPS were not taken into account sufficiently in previous work. Therefore, the research gap is identified as follows.

Research gap: The information and method for identification, representation, and prioritisation of TD in aPS are insufficient. In particular, first, an improvement of the TD classification to support TD identification is needed. Second, additional research on graphical modelling to represent TD is required. Third, a risk assessment method for a specific type of TD allowing extensibility, flexibility and automated analysis does not exist. Fourth, information about benefits, interests, sources and awareness of TD in the aPS domain is not well reported.

In the following chapter, the research method is presented.

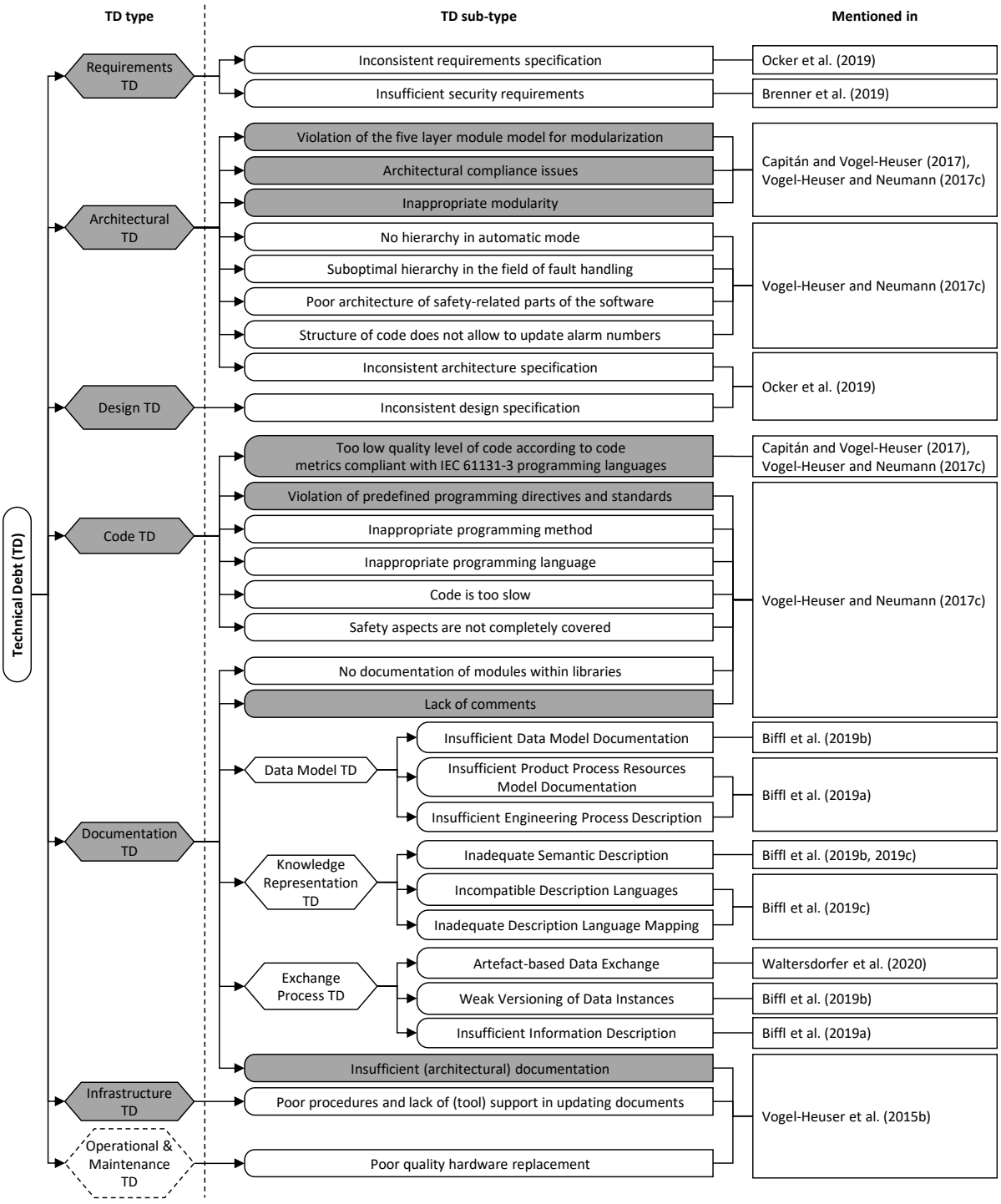


Figure 6: State-of-the-art TD classification for aPS based on classification of Li et al. (2015) (white boxes: enlargement; dashed-line boxes: TD types for aPS introduced in Vogel-Heuser and Rösch (2015a); grey boxes: existing TD (sub-)types according to Li et al.) (adapted from Capitán and Vogel-Heuser (2017) and Vogel-Heuser and Neumann (2017c)).

Table 2. Evaluation of relevant work (sorted into working groups) based on rating scheme in Table 1.

Work \ Requirement		***Req-Classification***	***Req-Modelling***	***Req-Risk***	***Req-Survey***
Avgeriou	(Li et al., 2015)	+ Classical SW	○ LC	-	+ Literature
	(Digkas et al., 2021)	-	-	○ GA	○ Classical SW
Dell'Olmo	(Franco, 2020)	-	○ Strategic level model	○ GA	-
Bosch, Tichy & Martini	(Martini et al., 2015), (Martini and Bosch, 2016, 2017)	○ Architectural TD	○ LC	-	○ Embedded SW
	(Besker et al., 2017)	○ aPS SW	-	○ LC	○ LS
Weippl	(Brenner et al., 2019)	○ aPS SW security	-	○ LC	-
Biffl & Lüder	(Biffl et al., 2019a)	○ Data exchange	-	+ Tabular approach	○ LS
	(Biffl et al., 2019b, 2019c)	○ Data exchange	+ Preliminary cause-effect model	-	○ LS
	(Waltersdorfer et al., 2020)	○ Data exchange	-	-	+ Tiers not considered
TUM AIS	(Vogel-Heuser and Rösch, 2015a), (Vogel-Heuser et al., 2015b)	○ Coarse-grained	-	-	○ LS
	(Vogel-Heuser and Neumann, 2017c)	○ aPS SW	-	-	○ LS
	(Capitán and Vogel-Heuser, 2017)	○ aPS SW	-	+ Modularity assessment	○ LS
	(Ocker et al., 2019)	○ Design TD	-	+ Ontologies approach	-
	(Vogel-Heuser et al., 2021)	+ Mechatronics	-	○ LC	+ Mechatronics
	(Bi et al., 2021, 2022)	+ Mechatronics	-	○ LC	+ Mechatronics
	(Vogel-Heuser et al., 2020a, 2022a)	-	+ TD not considered	○ LC	+ Mechatronics

SW: software; ***LC***: limited consideration (only mentioned, no method proposed); ***GA***: general assessment (assessing risk from TD in general, not focusing on specific TD type); ***LS***: limited survey (from laboratory plant, personal experience or one company)

3. Research activities

The work program is structured into three parts: (1) data collection, (2) data analysis, and (3) implementation and evaluation. The relationship between research problems, research activities (RAs) and publication is presented in Figure 7. In the following, the research activities are described in detail.

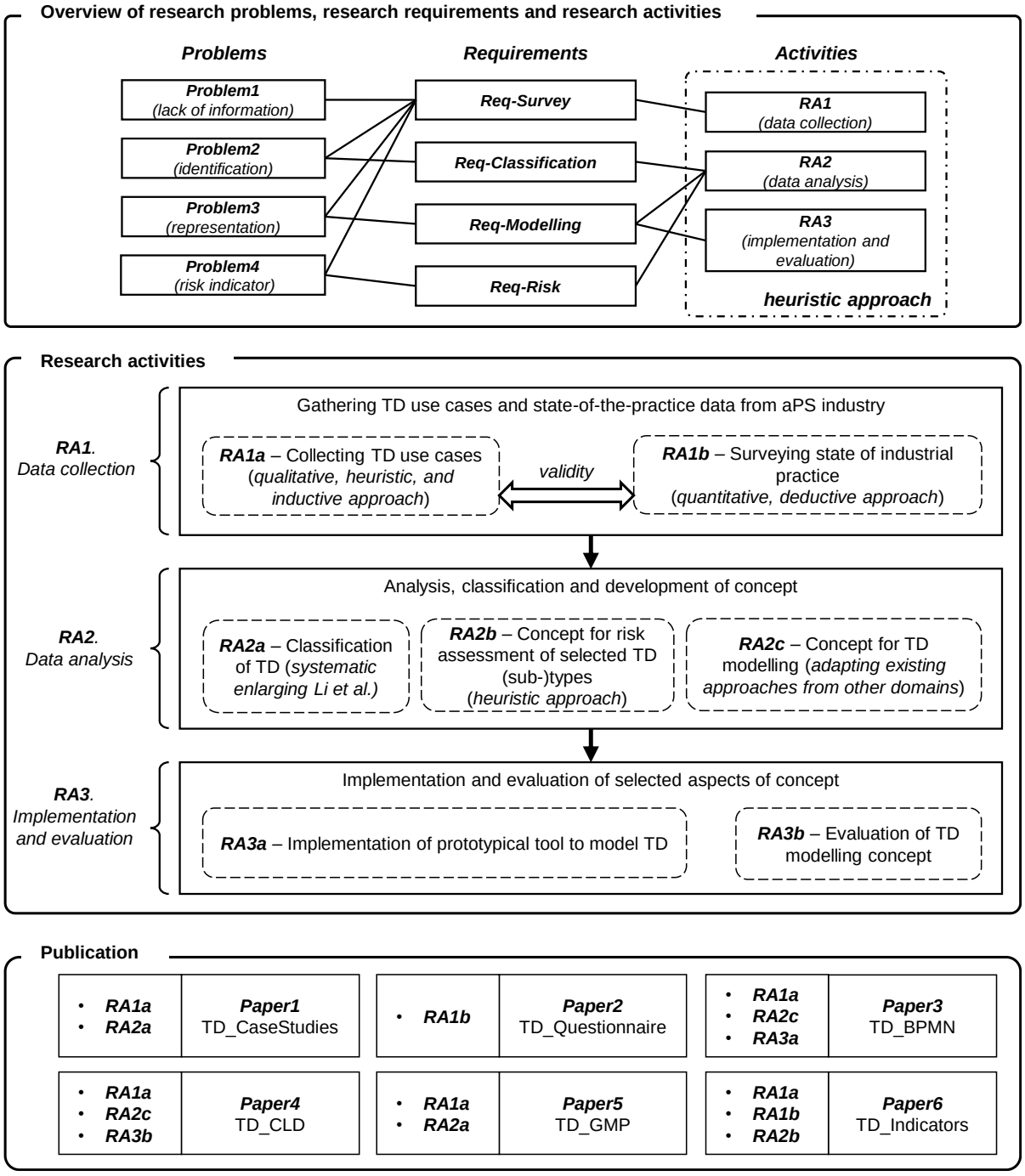


Figure 7: Research problems, activities and publication (namely paper numbers as occur).

3.1. Data collection (*RA1*)

This RA aims to address ***Req-Survey*** by gathering concrete TD cases and state of industrial practice from the aPS industry. The work started with an inductive investigation (cf. ***RA1a*** in Figure 7) and a deductive study (cf. ***RA1b*** in Figure 7). The inductive investigation was conducted with case studies, while the deductive approach used an online survey to evaluate the hypothesis grounded on the findings of the inductive approach. Selected case studies from companies were conducted to meet practical industrial requirements and theoretical contributions. Triangulations are employed to increase validity. Examples of triangulation are: (1) conducting interviews with people from multiple disciplines and roles and companies from multiple industrial sectors (data triangulation) or (2) using an online survey besides interviews (methodological triangulation).

3.1.1. Collecting TD use cases (*RA1a*)

An essay questionnaire was developed and gradually improved during the research. The essay questionnaire was intended for interviews with answers in a long-text format to collect TD data from different disciplines. The interviews examine:

- Software: (1) TD from mechanical or electrical engineering, which induces undesired TD to software design and implementation, and (2) methods for TD recovery (e.g. module revisions or versions).
- Electrics/electronics: TD taken during the cabinet design and start-up phase, which induces TD in software or mechanics.
- Mechanics: TD taken at the design phase causing long-term extra costs to the maintenance phase, such as the high effort required to visually monitor the machine for troubleshooting an issue or for conducting maintenance activities.

The activity follows the heuristic approach with a focus on the following aspects:

- TD at other development phases, such as the manufacturing phase (e.g. electrical cabinet construction) and assembly phase (e.g. assembly of mechanical and/or electrical components and software integration).
- TD at the commissioning phase, which features short deadlines that cause time pressure, and are a common trigger of TD, as aforementioned.

These focused phases are development steps where TD is most likely to occur. The aimed deliveries of ***RA1a*** are:

- Concrete TD occurrences from different disciplines at different aPS development phases. This activity is also a data preparation step for upcoming activities.
- Qualitative list of sources for TD and existing methods for TD recovery.

3.1.2. Surveying state of industrial practice (*RA1b*)

For the online survey, a multiple-choice questionnaire was developed based on the existing questionnaire introduced by Vogel-Heuser et al. (2017a). The existing questionnaire, which focused on software maturity, was extended with TD aspects. Quantitative results from ***RA1b*** are used as proof for the qualitative results from ***RA1a***. In addition, since MM and PM might follow different development practices (cf. section 2.6.1), it has to be explored how selected aspects affect TD: (a) MM and PM, (b) project characteristics, and (c) different reusability and modularity levels.

The aimed deliveries of ***RA1b*** are:

- State of industrial practice of TD and its management in the aPS domain focusing on benefits, interests, sources and awareness of TD.

3.2. Data analysis (*RA2*)

This RA generalises, classifies, prioritises and represents TD items collected in ***RA1a*** enlarging Li et al. (2015) classification. This activity aims to address ***Req-Classification***, ***Req-Modelling*** and ***Req-Risk***.

3.2.1. Classification of TD (*RA2a*)

Concrete TD items from ***RA1*** are analysed to establish a classification for TD based on the classification of Li et al. (2015). A synthesis across cases is conducted to gradually build a body of knowledge from individual TD examples.

The aimed deliveries of ***RA2a*** are:

- The enlargement of current TD classification for the aPS domain. It should be noted that ***RA1a*** follows a heuristic approach focusing on selected aspects (cf. section 3.1.1);

thus, the completeness of the classification from this work is not claimed and only selected TD areas are investigated.

3.2.2. Concept for risk assessment of selected TD (sub-)types (*RA2b*)

Once TD (sub-)types are classified, indicators for selected classes are developed to support companies in assessing TD. The indicators support the practitioners in identifying upcoming TD items in the early phases and thus help to prevent some TD. For instance, the proposed modularity metrics in Vogel-Heuser et al. (2018) or complexity metrics in Fischer et al. (2021) could be adapted to indicate the low maturity level of other aspects of control software, such as control code documentation. This activity follows the heuristic approach since it has to study the distinct parameters of factors influencing activities related to the selected classes. The proposed concept is based on the GAMP method, which allows extensibility for additional factors that could be added to different calculation stages. Adhering to the categories of metrics presented by Hristov et al. (2012) in the classical software engineering domain, a list of possible indicators is explored for control code documentation artefacts in the aPS domain.

The aimed deliveries of ***RA2c*** are:

- A proposal of risk assessment concept for selected TD (sub-)types.

3.2.3. Concept for TD modelling (*RA2c*)

Once upcoming TD is identified in the beginning, for example, by proposed metrics in ***RA2b***, the following research activity is to answer how aspects of TD, e.g. sources and effects, can be modelled. The results from this RA are the necessary inputs for ***RA3***.

There is a need to model TD at different levels, e.g. from strategic to implementation. Besides TD in the software discipline, the model should include cross-disciplinary TD from mechanical and electrical disciplines and throughout the life cycle of aPS. In addition, the connections between related TD items in different disciplines should be annotated in the model. Thus, this research activity adapts existing modelling approaches from other domains. First, to illustrate the propagation of TD aspects, e.g. adverse effects, across the life-cycle phases, BPMN is selected as it allows the extendibility of graphical representation for specifying elements along a workflow, following the idea in Vogel-Heuser et al. (2020a). Second, CLD is selected following the idea of Franco (2020) in the classical software engineering domain, which could be extended to model TD artefacts in other disciplines, e.g. mechanical or electrical. Also, CLD

enables loop dominance analysis, a method in system dynamics, to derive trends of TD (a variable in CLD model) over time.

The aimed deliveries of ***RA2c*** are:

- Modelling methods for selected aspects of TD, such as sources and effects.

3.3. Implementation and evaluation (*RA3*)

At this stage, the usability of the selected concept is evaluated to address ***Req-Modelling***. It should be noted that the activity only addresses selected aspects of TD modelling with BPMN and CLD.

3.3.1. Implementation of prototypical tool to model TD (*RA3a*)

Within this RA, a prototypical implementation of TD adaptation for BPMN is developed to describe a technical compromise and its propagation in a process model. The tool assists in a better representation of the TD use cases, e.g. sources and contributing factors to TD, to support engineers or technicians in illustrating the cases to the management team for further action, e.g. prioritising repayment for the technical shortcuts.

3.3.2. Evaluation of TD modelling concept (*RA3b*)

This activity investigates the usability of the CLD+TD concept by surveying industrial experts.

The aimed deliveries of ***RA3*** are:

- Prototypical tool for BPMN+TD modelling with graphical notations to specify selected aspects of TD, such as sources and effects.
- Experts' feedback on the usability of CLD+TD modelling, e.g. in depicting effects of TD or in illustrating trends of TD, and improvement potentials.

In the next chapter, the publications are described.

4. Summary of the publication

This chapter highlights the research results obtained during the work reported in six papers (four published, one accepted, and one submitted) on which this dissertation is based. All published papers are listed in Scopus or Web of Science databases. The individual contribution of each paper is described in Table 3. The included papers (namely paper numbers as they occur) are attached in the Appendix.

Table 3: Overview of the candidate's individual contribution (for each activity of each paper, the contribution of all authors is 100%).

	Conceptualization	**Data collection**	**Elaboration**	**Writing**
Paper1-TD_CaseStudies [*published*]	10%	20%	50%	70%
Paper2-TD_Questionnaire [*published*]	20%	20%	60%	80%
Paper3-TD_BPMN [*published*]	30%	30%	50%	70%
Paper4-TD_CLD [*published*]	70%	50%	80%	80%
Paper5-TD_GMP [*submitted on 18 March 2022*]	30%	0%	60%	80%
Paper6-TD_Indicators [*accepted on 04 May 2023*]	10%	30%	60%	60%

Notes:

- **Conceptualization**: Development and conceptual design of the research project
- **Data collection**: Gathering, collection, acquisition or provision of data, software or sources
- **Elaboration**: Analysis/evaluation or interpretation of data, sources and conclusions drawn from them
- **Writing**: Drafting of the manuscript

4.1. Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey (*Paper1*)

Quang Huan Dong and Birgit Vogel-Heuser

Summary of the paper

The surrounding factors and their impacts of TD on the software development aspect of aPS are studied. Besides the mechanical, electrical and software engineering aspects, the investigated aPS also include pneumatic or hydraulic aspects. The employed research method

is TD4aPS (Technical Debt for aPS), which is adapted from the SWMAT4aPS (Software Maturity for aPS) method presented in (Vogel-Heuser et al., 2017a). The study was conducted with the participating MM and PM in Germany to collect qualitative TD data from different contexts. Two case studies are collected from companies in Tier 2 (MM and PM). Case A is from a MM, and case B is from a PM. Case A represents a TD taken during the design phase of the examined aPS. Case B represents a TD that arises in the commissioning phase due to time pressure.

The reported TD items show two typical characteristics of TD in the aPS domain. In case A, as the TD from the mechanics discipline induces adverse effects on the software discipline, the TD can be regarded as a cross-disciplinary TD. In case B, as the effects of TD spread to different life-cycle phase(s), the TD can be regarded as a cross-life-cycle-phase TD. The identified TD items are categorised according to the systematic classifications introduced by Li et al. (2015) and Vogel-Heuser and Rösch (2015a).

Based on the results of an online survey, a preliminary evaluation was conducted to examine the qualitative findings collected from the case studies. This evaluation was intended to enhance the study's validity. Early results indicated that the largest additional long-term effort occurred in the software discipline and time pressure was the most frequent reason for the choice of TD. Thus, the quantitative results from the online survey seem to confirm the qualitative results of the case studies. Nevertheless, additional analysis is necessary to validate the achieved findings.

Individual contribution of the candidate

My contribution to this paper (Dong and Vogel-Heuser, 2018) includes the literature review and questionnaire preparation. I conducted the investigation by analysing the study data and visualising the findings. The initial version manuscript was written by me.

This section was published as ***Paper1***:

> Dong, Q. H., and Vogel-Heuser, B. (2018). Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey. *IFAC-PapersOnLine: 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM), 51*(11), pp. 1192-1199. Elsevier. https://doi.org/10.1016/j.ifacol.2018.08.428

4.2. Technical Debt as indicator for weaknesses in engineering of automated production system (*Paper2*)

Quang Huan Dong, Felix Ocker and Birgit Vogel-Heuser

Summary of the paper

Through the online survey, insights on TD in 48 German MM and PM companies working with aPS are obtained. The state of industrial practice on selected aspects such as benefits, interests, sources and awareness of TD in the aPS domain are reported.

On average, PM has a more significant project size and has better TD awareness than MM. Customer-specific projects suffer more from TD interest, general projects have less TD benefit than other kinds of projects, and general projects have better TD awareness. When the electrical/electronics (E/E) discipline takes TD benefit, 43% of TD interest comes to the software discipline. Notably, in 33% of projects in which E/E engineers are the majority, E/E engineers do force TD's interest in software discipline. Although software development/engineering is the most affected discipline by TD interest, it is also the most beneficial from taking TD. Among the surveyed sources of TD, time pressure received the most responses, confirming the findings of earlier work such as Martini et al. (2015) in the domain of embedded systems.

The results also indicated that different modularity levels lead to different reusability levels, and different levels of reusability result in different TD amounts. Specifically, high reusability results in low TD interest.

It should be noted that with the online survey method, the result could only provide an overview of top-level aspects of the state of industrial practice regarding TD in aPS. Recently, more profound results have been presented in two studies by Vogel-Heuser et al. (2021) and Bi et al. (2021) with the interview method. However, since these two studies focused on mechatronic products, there still needs a further investigation on the state of industrial practice of TD in aPS – a subset of mechatronic products, as aforementioned.

Individual contribution of the candidate

My contribution to this paper (Dong et al., 2019) includes the literature review and questionnaire preparation. I conducted the investigation by analysing the survey data and visualising the findings. The initial version of the manuscript was written by me.

This section was published as ***Paper2***:

> Dong, Q. H., Ocker, F., and Vogel-Heuser, B. (2019). Technical Debt as indicator for weaknesses in engineering of automated production systems. *Production Engineering*, *13*(3), pp. 273-282. Springer. https://doi.org/10.1007/s11740-019-00897-0

4.3. Modelling technical compromises in electronics manufacturing with BPMN+TD – an industrial use case (*Paper3*)

Quang Huan Dong and Birgit Vogel-Heuser

Summary of the paper

Some manufacturing plants are relocated because of new market opportunities, labour costs, or material changes. The plants are often relocated from Germany (Europe) to Asia depending on the product margins or benefits to allow profitable business. On this subject, a field study is conducted in Singapore in the domain of electronics products for automation. The surveyed companies belong to Tier 3 (device or platform manufacturers). Early results indicate that some TD due to time pressure during the transfer of the manufacturing systems might be needed to start production as early as possible. This work proposes an initial BPMN enlargement for TD, namely BPMN+TD, in order to model some selected aspects of the collected TD example.

Two extended graphical notations, namely TechnicalDebtTask and TechnicalDebtItem, are supported in a basic version of the BPMN+TD prototypical tool. The tool enhances the BPMN2 Modeler project, an open-source Eclipse plugin. BPMN+TD extends the BPMN2 Modeler with two proposed TD graphical notations above. The description of BPMN+TD is in XML format, which can be generated by the tool automatically. The generated XML file obtained from the BPMN+TD tool can be imported into a BPMN engine to create and execute instances of the defined process model. Thus, the activities related to a TechnicalDebtTask or TechnicalDebtItem can be assigned to responsible staff or disciplines. Therefore, changes related to the TD can be tracked. New information added to the BPMN+TD model could help to prevent knowledge loss in case the related personnel leave the company.

It should be noted that the survey results presented in ***Paper2*** reported a relatively low TD awareness in German aPS manufacturers. However, this exploratory case study shows quite good TD awareness at this electronics manufacturer.

Individual contribution of the candidate

My contribution to this paper (Dong and Vogel-Heuser, 2021a) includes the literature review, questionnaire preparation, and interview documentation. I conducted the investigation by performing the transcription and analysis of the interview data, visualising the findings and developing the prototypical tool. The initial version of the manuscript was written by me.

This section was published as ***Paper3***:

> Dong, Q. H., and Vogel-Heuser, B. (2021a). Modelling technical compromises in electronics manufacturing with BPMN+TD – an industrial use case. *IFAC-PapersOnLine: 17th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, *54*(1), pp. 912-917. Springer. https://doi.org/10.1016/j.ifacol.2021.08.108

4.4. Modelling Industrial Technical Compromises in Production Systems with Causal Loop Diagrams (*Paper4*)

Quang Huan Dong and Birgit Vogel-Heuser

Summary of the paper

A concept, namely CLD+TD, is proposed to model some causal aspects of TD in aPS with CLD. Following the TD definition by Li et al. (2015), three main factors, including the benefit of TD, the interest of TD, and the type of TD, are formulated. Due to the interdisciplinary characteristics of aPS, one TD item might induce another. Thus, an additional association arrow is introduced to represent TD inducing or injecting another TD. Therefore, CLD+TD includes four graphical notations to illustrate the above factors: (1) three adjustments of the original CLD factor box for the interest of TD, the benefit of TD, and the type of TD; and (2) TD-inducing arrow.

As proof of concept, CLD+TD is applied to model some TD use cases collected in the commissioning phase of an industrial company working in automation. The findings show the cross-company or cross-tier characteristic of TD. In particular, the presented use cases reported the significant influence that the purchasing departments of Tier 1 (customers producing final products) and Tier 2 (MM or PM) have on decisions to acquire TD in automation, influencing all participating tiers.

Following the approach of Haraldsson (2004), a loop dominance analysis is conducted on a reported CLD+TD model to determine RBP in order to explain the circular trend of TD, i.e. TD increase and decrease circularly.

Individual contribution of the candidate

My contribution to this paper (Dong and Vogel-Heuser, 2021b) includes conceptualization, literature review, questionnaire preparation, and interview participation. I conducted the investigation by performing transcription, analysing the interview and evaluation data, and visualising the findings. The initial version manuscript was written by me.

This section was published as ***Paper4***:

> Dong, Q. H., and Vogel-Heuser, B. (2021b). Modelling Industrial Technical Compromises in Production Systems with Causal Loop Diagrams. *IFAC-PapersOnLine: 4th IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT), 54*(4), pp. 212-219. Springer. https://doi.org/10.1016/j.ifacol.2021.10.036

4.5. Including validation of process control systems' engineering into the Technical Debt classification (*Paper5*)

Quang Huan Dong and Birgit Vogel-Heuser

Summary of the paper

The associated risks with TD may be critical if TD is overlooked or not fixed properly. For instance, the product might not be used safely (in the healthcare supplies industry) or be contaminated (in the food and beverage sector) due to a design fault which is not adequately addressed. The work extends the TD classification for process automation systems that need to undergo a validation process according to GAMP, which is required for pharmaceutical production or food processing, for example.

A meta-analysis was conducted with engineering documents provided by a world-leading machine and plant manufacturer which had been adopting V-Model and GAMP. The functional specifications followed the V-Model process, and the risk evaluation was in accordance with GAMP. The compositions described engineering and risk in electrical, software, and

mechanical disciplines. The risk trace matrix document also includes feedback from the participating company's customer (Tier 1).

The solution mentioned in each risk entry was assessed to check whether there was a suboptimal solution; thus, TD could be detected. Two workflows with different approaches were designed for the study to achieve a TD classification. The two methods supported the researchers in systematically reviewing and identifying TD in significant long documents (more than a hundred pages). On the one hand, the bottom-up approach focused on discrimination between implementation and guidelines/standards. On the other hand, the top-down approach analysed risk entries.

TD identification was conducted by comparing the identified guidelines with the implementations. Some selected TD use cases confirmed by industry experts are presented. In the first example, there was a discrepancy between risk assessment guidelines and practice. The second example was a discrepancy between practice and GAMP. Regarding example 3, there was an intentional judging risk as a low priority, although the risk is high and other options are available (e.g., software bug).

The presented meta-analysis approach on engineering documents can be employed as a TD identification method for such process control systems. The TD classification for process automation was thereby enlarged for a specifically critical type of machine that may harm the health of many humans.

Individual contribution of the candidate

My contribution to this paper (Dong and Vogel-Heuser, 2022) includes the literature review, evaluation preparation, analysing the study data, and visualising the findings. The initial version of the manuscript was written by me.

This section was submitted as ***Paper5***:

> Dong, Q. H., and Vogel-Heuser, B. (2022). Including validation of process control systems' engineering into the Technical Debt classification. *Forschung im Ingenieurwesen/Engineering Research.* Springer. *(submitted on 18 March 2022)*

4.6. Semi-automatic assessment of lack of control code documentation in automated production systems (*Paper6*)

Quang Huan Dong, Birgit Vogel-Heuser and Eva-Maria Neumann

Summary of the paper

Software functionalities of long-lifetime aPS must be enlarged to produce new or more complex products; thus, the complexity increases. In this case, documentation TD might hinder source code readability or misinterpretation. This is due to high complexity or poor control code documentation that might hinder software maintainability, which might introduce undesired additional costs. In the classical software engineering domain, various approaches have been introduced to measure software characteristics such as complexity or code documentation (e.g. code comment). However, a method is lacking to determine the criticality or quality of documentation in the aPS domain.

A study is conducted to analyse the documentation aspect of the results of a survey. The survey addressed a German-speaking community (and) involved mechanical-, embedded systems-, and software engineering disciplines in MM and PM. The analysis indicates that (1) document availability is poor in early engineering phases, and it later varies at different disciplines or company sizes, (2) the detail level of in-house guidelines differs at disciplines, (3) automatic generation of engineering documents is still poorly applied at disciplines, and (4) the usage of community guidelines is less than the usage of company-specific guidelines or checklists.

A set of factors influencing documentation activities is derived based on the survey results and state of the art. The focus is on the control code documentation. The work employs the identified factors and the GAMP method in formulating a risk-based concept to indicate documentation TD, namely RPI4DD. The RPI4DD approach is extensible, flexible, and can be (semi-)automated; therefore, feasible for industrial large-scale control software projects, which often involve hundreds of POUs. Hence, integrating RPI4DD into the risk management process will support practitioners in identifying and analysing the risks associated with control code documentation TD in technical systems; therefore, earlier reaction or risk-reduction measures can be developed to enhance the system quality.

Individual contribution of the candidate

My contribution to this paper (Dong et al., 2023) includes the literature review, elaboration, proof of concept, visualising the findings, and drafting the manuscript.

This section was accepted as ***Paper6***:

> Dong, Q. H., Vogel-Heuser, B., and Neumann, E.-M. (2023). Semi-automatic assessment of lack of control code documentation in automated production systems. *at – Automatisierungstechnik*. De Gruyter. https://doi.org/10.1515/auto-2022-0146 *(accepted on 04 May 2023)*

In the following chapter, the research contribution is discussed.

5. Discussion and outlook

To conclude the thesis, a summary of the research contribution is discussed. It relates the findings to the problems presented in the Introduction. Afterwards, it is reviewed if the requirements are fulfilled, and an outlook on future work is given.

5.1. Summary of contribution

The qualitative- and quantitative findings (primarily presented in the included papers) and the approach proposed in this work reveal critical details of undesired costs among disciplines along life-cycle phases from a TD perspective in the aPS domain. The enlarged TD classification is provided to contribute to the efforts to support TD identification – the first activity in TD management – in the aPS domain. In addition, embedding TD perspective into modelling methods can assist practitioners in presenting TD to involved stakeholders. Furthermore, the proposed risk assessment of TD in control code documentation can support the practitioners in identifying upcoming TD items in the early phases. Thus, some TD can be prevented and therefore, unscheduled maintenance efforts can be reduced.

The work investigates the TD characteristics in different disciplines and phases of aPS. The proposed concept regarding indicators for the documentation of TD demonstrates how various metrics collected from different sources (e.g. static code analysis tools or test management tools) can be integrated into a model to support decision-making in evaluating the risk of TD in aPS maintenance. This work also supports the idea that integrating the TD concept into development and maintenance activities is a potential approach for reducing cost problems related to the aPS. Formulating graphical representations of identified TD during the development of aPS enables leveraging current knowledge in cross-disciplinary and cross-life-cycle-phase characteristics of TD in aPS. The thesis also enriches the literature on TD research in the aPS domain, which is scarce since there were only few publications on the topic primarily targeting TD in control software. Furthermore, the stepwise research procedure presented in section 3 can be employed in future work on the TD in aPS.

In the following, the research contribution regarding the four research problems which are initially stated in the Introduction chapter is discussed.

- ***Problem1***: *Lack of information about benefits, interests, sources and awareness of TD in the aPS domain.*

  Quantitative results on the current industrial practices and preferences regarding benefits, interests, sources and awareness of TD in the aPS domain are reported in ***Paper2***. The body of knowledge is enriched with findings on the decision to take TD, the spreading of TD interests among disciplines in their engineering timeline, and an overview of TD awareness at MM and PM.

  The presented results in ***Paper2*** indicate that the participating companies can quickly benefit from TD. However, the long-term cost of recovering from TD is significantly higher than the short-term costs saved. In addition, awareness of TD in these companies is low. Therefore, the aPS manufacturers should pay more attention to expenditures for TD. The developed questionnaire can be utilised as a self-assessment method for other companies to examine their situations with the benchmark findings of this work.

  It is worth noting that these findings in ***Paper2*** could only provide a limit and top-level results regarding the state of industrial practice of TD in aPS as the employed research method was the online survey. Nevertheless, combined with insights from case studies, e.g. characteristics of TD or models of relations between TD (sub-)types, these findings lay a groundwork for the broad-scale evaluation of TD in mechatronic products conducted by Vogel-Heuser et al. (2021) and Bi et al. (2021, 2022) to study, e.g. patterns of TD.

- ***Problem2***: *Insufficient classification to support TD identification in the aPS domain.*

  The management of TD in aPS demands substantial knowledge and experience on the TD artefacts. The first step of TD management is awareness, i.e., identifying TD by studying its causes. Qualitative results are achieved with TD use cases reported in ***Paper1***, ***Paper3***, ***Paper4*** and ***Paper5*** collected from industrial companies working with aPS. Following the systematic classification of Li et al. (2015), the body of knowledge is enlarged with new TD (sub-)types categorized into the aPS life-cycle phases based on the initial phases and causes of TD. Hence, these details can support practitioners in correctly and quickly identifying TD in similar cases.

  A synthesis across publications is conducted and presented in Figure 8 to provide an overview of the identified TD which is individually reported in the included papers.

This work enlarges the current TD classification for aPS (cf. section 2.6.3) with a new TD type, namely *Industrial engineering TD*. In addition, new TD sub-types are found in *Manufacturing TD* and *Commissioning TD* groups, which are understudied in earlier work. Furthermore, new TD sub-types in the risk assessment process are also explored.

Two typical characteristics of TD in the aPS domain, i.e. cross-disciplinary and cross-life-cycle-phase, are illustrated in Figure 8. TD use cases with cross-disciplinary and cross-life-cycle-phase factors are reported in ***Paper1*** and ***Paper4***. TD use cases presented in ***Paper4*** show the contagious characteristic of TD spanning from strategic to implementation phases of aPS. An example of a TD spanning across sites of the manufacturers is described in ***Paper3***. These findings assist practitioners in understanding and predicting the effects of TD in aPS.

Findings in ***Paper5*** highlight the contagious characteristic of TD in aPS. TD may not only spread throughout the life cycle of aPS but may also burden the operating company and customer consuming the final products, e.g., yoghurt or a pharmaceutical product. Although Avgeriou et al. (2016b) stated that "*technical debt presents an actual or contingent liability whose impact is limited to internal system qualities, primarily maintainability and evolvability*", these findings provide empirical evidence that the TD impact could reach further than just internal system qualities. In addition, since the presented TD use cases show risks and adverse effects on the aPS in the healthcare sector, the findings are critical as they could help to avoid harming lives.

It is worth noting that due to the heuristic approach to finding TD, this work can only identify some selected aspects; thus, there is no claim of the completeness of the classification. The selection of TD use cases reported is based on an accessible sampling. Despite this limitation, it should be noted that all informants were professionals with extensive experience in the automation field. Another limitation is that the research was only performed with aPS working in some industries, such as automotive, pharmaceutical or healthcare. Hence, the findings may not be generalised to other aPS sectors. Nevertheless, the collected use cases were from different companies of different sizes (i.e. small-and-medium, large and very large organisations) and different tiers, which could provide a decent degree of generalizability.

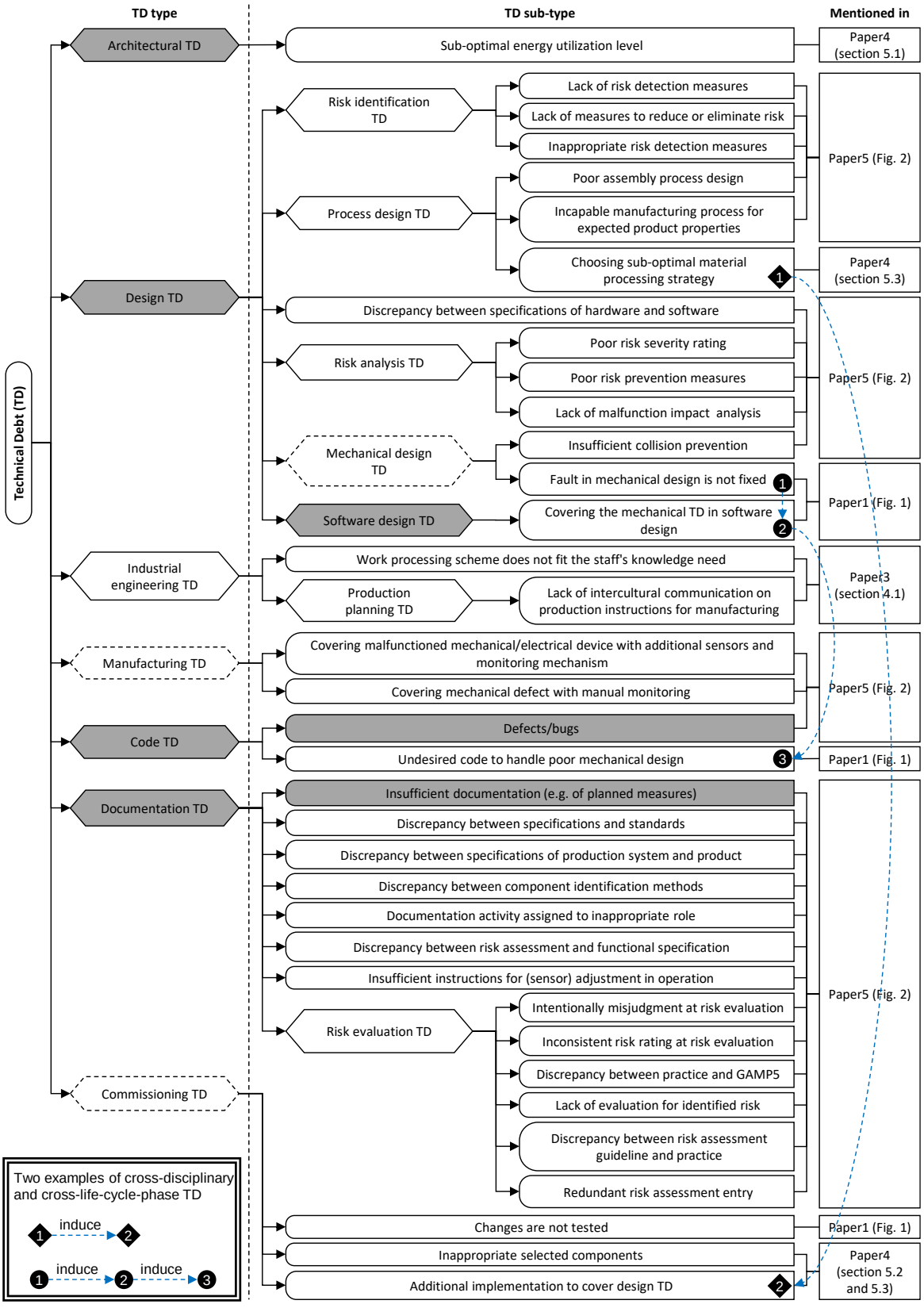


Figure 8: Enlargement of TD classification for aPS following Li et al. (2015) systematic classification (white boxes: enlargement; dashed-line boxes: TD (sub-)types introduced in Vogel-Heuser and Rösch (2015a); grey boxes: TD (sub-)types according to Li et al.) (adapted from Dong and Vogel-Heuser (2018, 2021a, 2021b and 2022)).

- ***Problem3***: *Lack of capable methods to represent TD workflow in the aPS domain.*

  Two methods are proposed to assist practitioners in modelling the cross-disciplinary and cross-life-cycle-phase characteristics of TD in the aPS domain. First, the work focused on annotating TD in a process model with the BPMN+TD approach reported in ***Paper3***. Second, based on Systems theory, the proposed CLD+TD approach in ***Paper4*** aims at modelling the TD artefacts with the concept of feedback. The presented novel prototypical hierarchy view could support illustrating the relationship between factors of the related TD use cases in a 3D view. There, the CLD+TD models could be displayed at different layers (e.g., from strategic to implementation). This view could enhance the representation of the synthesised model. Thereby, the root cause and hierarchy influence of the TD factors can be described in the model relatively quickly.

  Compared with the former approaches, BPMN+TD and CLD+TD support engineers or technicians to represent some TD aspects explicitly, such as involved disciplines, which was not addressed in (Biffl et al., 2019b) (Martini and Bosch, 2017). In addition, the CLD+TD approach supports explaining the trend of TD in the crisis point model, which was just partially verified by Martini et al. (2015).

- ***Problem4***: *Inadequate approach to evaluate TD's risk in control code documentation of aPS.*

  Quantitative results on the current industrial practices and preferences in ***Paper6*** indicate a lack of documentation in the aPS domain. The presented RPI4DD concept supports the engineers in measuring the risk of TD in control code documentation in control software, which is a prerequisite for TD identification and prioritization activities in TD management. Therefore, the work contributes to the efforts to improve documentation quality in the aPS domain; thus, engineering artefacts (e.g. complex code) could be understood quicker. Hence, the time to develop aPS could be reduced, thus, low costs.

  Compared with the former approaches, RPI4DD follows a systematic approach, i.e., factor hierarchy, which allows an adaption of the proposed concept methodically. The related work on TD risk assessment approaches, e.g., Biffl et al. (2019a) or Digkas et al. (2021), did not present concrete steps to systematically enlarge or adapt the approaches, e.g., to measure the risk of different TD types. Besides documentation, the

proposed approach can be generally applied to other aspects of control software quality (e.g., testing) or could be further applied to other disciplines.

Above all presented findings, the enlargement of TD classification shown in Figure 8 is considered the main contribution of this thesis. In the following, the fulfilment of the stated requirements is discussed.

5.2. Fulfilment of the requirements

A self-assessment of the fulfilment of the requirements is conducted and presented in Table 4. Significant concerns might arise as there is still a requirement partially fulfilled. However, the three fulfilled requirements show the positive outcome of this work. Due to the heuristic approach, only selected aspects of TD in aPS are addressed in this work. Nevertheless, the work still follows the design science research approach. For instance, industrial studies were conducted to collect TD use cases, followed by proposing concepts to model the collected TD use cases and evaluating a selected concept's usability.

Although ***Req-Modelling*** is just partially fulfiled, both proposed methods BPMN+TD and CLD+TD allow abstraction by simplifying and annotating some selected aspects of a TD use case with graphical and textual notation. Since the proposed methods are adapted from BPMN and CLD, which are well-known modelling methods, thus, the BPMN+TD and CLD+TD models could be interpreted by stakeholders with trivial training (interpretation). Regarding executability, the generated XML file from the BPMN+TD model with the prototypical tool can be imported into a process execution engine. The source code of the tool is published online (Dong and Vogel-Heuser, 2023). In addition, the presented CLD+TD models can be simulated using system dynamics modelling and simulation software application such as Simantics System Dynamics (Simantics Team, 2022).

To fully address ***Req-Modelling***, future work can include more aspects (e.g., additional TD notations or factors) to enhance the results since the presented approaches in ***Paper3*** and ***Paper4*** are extensible.

Table 4: Summary of the rating of requirement fulfilment (+ fulfilled, ○ partially fulfilled, - not fulfilled).

Requirement	Rating	Detailed Rating and Reference to Publication
Req-Classification – Improved granularity of TD classification with convenience sample at aPS life-cycle phases.	+	Fulfilled – An enlargement of TD classification (cf. Figure 8) is reported in ***Paper1***, ***Paper3***, ***Paper4***, and ***Paper5***. The work considers TD in different disciplines and life-cycle phases of aPS. As this work is exploratory and heuristic, there is no claim of the completeness of the classification, but selected aspects of TD in the aPS domain were reported.
Req-Modelling – Support of modelling TD use cases considering discipline and life-cycle-phase aspects to illustrate the propagation of TD.	○	Partially fulfilled – ***Paper3*** presents a BPMN adaptation, i.e., BPMN+TD, to annotate selected aspects of a TD use case in a process model. ***Paper4*** introduces the CLD+TD concept to model selected TD artefacts with CLD. Although the BPMN+TD and CLD+TD approaches are modelling methods with graphical notation, the requirements for abstraction, interpretation and executability are still not fully addressed yet. In addition, ***Paper3*** only reported early findings and initial enlargement of BPMN for TD; therefore, those results can be seen as complementary results of this thesis. Thus, further work is needed to incorporate more aspects of TD to build up comprehensive TD notations.
Req-Risk – Support of assessment of the risk aspect associated with a specific type of TD.	+	Fulfilled – ***Paper6*** addresses selected factors of (semi-)automatically identifying documentation TD and proposes a Risk Prioritisation Number of Documentation concept, i.e., RPI4DD. The applicability of RPI4DD on control code documentation TD in aPS is presented. Extensibility and flexibility are considered in the approach.
Req-Survey – Quantitative and qualitative industrial survey including companies at different tiers working with aPS.	+	Fulfilled – All included papers consider data from the industry, including companies from different tiers working with aPS. Multiple industrial TD use cases were reported.

A self-assessment of the assumptions for requirements is also conducted and presented as follows. First, after each interview, the TD classification has been gradually enlarged with new TD (sub-)types identified. The improved classification supported identifying TD more quickly at the subsequent interviews. Thus, the **Req-Classification** assumption – identifying TD more effectively with TD classification improved – held. Second, the BPMN and CLD adaptions were employed to model selected aspects of TD in some aPS development activities such as design, documentation or commissioning. Thus, the assumption the **Req-Modelling** assumption – applicability of graphical modelling to represent all types of TD – is possible to be held. Third, a set of factors influencing control code documentation was successfully derived to indicate the risk of documentation TD. Thus, the **Req-Risk** assumption – measuring the risk of a specific TD with the related aspects driving that TD – is possible to be held. Fourth, the results from the online survey seem to confirm the qualitative results of the case studies. Thus, the **Req-Survey** assumption – the validity of results from an online survey – is possible to be held. Therefore, the presented requirements for TD in aPS seem to be proper for future research.

5.3. Outlook

The awareness of TD in the aPS domain is still moderately low; thus, undesired costs for unnoticed TD. The cross-disciplinary and cross-life-cycle-phase characteristics of TD in aPS increase this problem. Since the proposed methods to model TD use cases could assist in documenting and presenting TD to different stakeholders in the surveyed companies, awareness of TD in aPS could be increased. Towards further steps in TD management (e.g., assisting management decisions on TD in aPS), additional research is required on two other aspects (industrial engineering and management), according to Panetto et al. (2019). There, directives for the identified TD could be developed.

This work explored TD in aPS operating in some sectors (e.g., industrial automation in healthcare). More exploration in other sectors of aPS is required to enlarge TD classification further and investigate how sector characteristics (e.g., types of automated manufacturing processes) influence TD. The difference in TD awareness at the surveyed electronics manufacturers in Singapore and the surveyed German aPS companies raises an interesting question:

- How are global aPS companies coping with TD?

Future work could study global aPS companies' characteristics, i.e. international aspects, to answer the above question. For example, early findings presented in ***Paper3*** indicate a significant misunderstanding of qualification of maintenance staff or operators when plants of the electronics manufacturers are relocated from Europe to Asia. Future research can be carried out with global automotive companies, which are building factories around the world to produce specific cars or branches for e.g. different markets. It can examine if those car-makers encounter any TD like the ones reported in ***Paper3***.

Although many articles discuss TD in the computer science domain, there were only some conference papers and no journal paper fully dedicated to TD in aPS before ***Paper2***. Due to a more rigorous review process, a journal paper is often seen as superior to a conference paper; thus, the acceptance of academic researchers on ***Paper2*** shows that the topic of TD in the aPS domain has scientifically gained more recognition. Hence, opportunities for further publications can be seen since there are many results of TD's work in computer science that could be transferred or adapted for aPS or other domains.

6. References

Albers, A., Dumitrescu, R., Gausemeier, J., Lindow, K., Riedel, O., and Stark, R. (Hrsg.) (2022). *Strategie Advanced Systems Engineering – Leitinitiative zur Zukunft des Engineering und Innovationsstandorts Deutschland*. München, Germany: acatech. Available at: www.advanced-systems-engineering.de (Accessed: July 2022).

Ampatzoglou, A., Ampatzoglou, A., Chatzigeorgiou, A., and Avgeriou, P. (2015). The financial aspect of managing technical debt: A systematic literature review. *Information And Software Technology*, *64*, pp. 52-73. Elsevier. https://doi.org/10.1016/j.infsof.2015.04.001

Argyris, C., and Schön, D.A. (1978). *Organizational learning: A theory of action research*. Reading, MA: Addision-Wesley Publishing Company.

Avgeriou, P., Kruchten, P., Nord, R. L., Ozkaya, I., and Seaman, C. (2016a). Reducing Friction in Software Development. *IEEE Software*, *33*(1), pp. 66-73. IEEE. https://doi.org/10.1109/MS.2016.13

Avgeriou, P., Kruchten, P., Ozkaya, I., and Seaman, C. (2016b). Managing Technical Debt in Software Engineering. *Dagstuhl Reports*, *6*(4), pp. 110-138. Dagstuhl Publishing. https://doi.org/10.4230/DagRep.6.4.110

Beck, K. (2003). *Test-driven development: by example*. Addison-Wesley Professional.

Besker, T., Martini, A., Bosch, J., and Tichy, M. (2017). An investigation of technical debt in automatic production systems. In *The XP2017 Scientific Workshops* (pp. 1-7). New York, NY, USA: ACM. https://doi.org/10.1145/3120459.3120466

Besker, T., Martini, A. and Bosch, J. (2019). Software developer productivity loss due to technical debt—A replication and extension study examining developers' development work. *Journal of Systems and Software*, *156*, pp. 41-61. Elsevier. https://doi.org/10.1016/j.jss.2019.06.004

Bi, F., Vogel-Heuser, B. and Xu, L. (2021). Frequency and Impact of Technical Debt Characteristics in Companies Producing Mechatronic Products. In *2021 IEEE/ACM International Conference on Technical Debt (TechDebt)* (pp. 26-35). IEEE. https://doi.org/10.1109/TechDebt52882.2021.00012

Bi, F., Vogel-Heuser, B., Huang, Z., and Ocker, F. (2022). Characteristics, Causes, and Consequences of Technical Debt in the Automation Domain. *SSRN Electronic Journal*, *2*(3), pp. 54-59. Elsevier. https://doi.org/10.2139/ssrn.4299857 (*pre-print*)

Biffl, S., Kathrein, L., Lüder, A., Meixner, K., Sabou, M., Waltersdorfer, L., and Winkler, D. (2019a). Software Engineering Risks from Technical Debt in the Representation of Product/ion Knowledge. In *The 31st International Conference on Software Engineering and Knowledge Engineering* (pp. 693-700). KSI Research Inc. and Knowledge Systems Institute Graduate School. https://doi.org/10.18293/seke2019-037

Biffl, S., Ekaputra, F., Luder, A., Pauly, J., Rinker, F., Waltersdorfer, L., and Winkler, D. (2019b). Technical Debt Analysis in Parallel Multi-Disciplinary Systems Engineering. In *The 45th Euromicro Conference On Software Engineering And Advanced Applications (SEAA)* (pp. 342-346). IEEE. https://doi.org/10.1109/seaa.2019.00059

Biffl, S., Luder, A., Rinker, F., Waltersdorfer, L., and Winkler, D. (2019c). Quality Risks in the Data Exchange Process for Collaborative CPPS Engineering. In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)* (pp. 1217-1224). IEEE. https://doi.org/10.1109/INDIN41052.2019.8972322

Birkhofer, R., Feldmeier, G., Kalhoff, J., Kleedörfer, C., Leidner, M., Mildenberger, R., Mühlhause, M., Niemann, J., Schrieber, R., Wickinger, J., Winzenick, M., and Wollschlaeger, M. (2010). *Life-Cycle-Management Für Produkte Und Systeme Der Automation - Ein Leitfaden Des Arbeitskreises Systemaspekte Im ZVEI Fachverband Automation*. Frankfurt: Zentralverband Elektrotechnik- und Elektronikindustrie e.V.

Bocciarelli, P., D'Ambrogio, A., Giglio, A. and Paglia, E. (2017). A BPMN extension for modeling cyber-physical-production-systems in the context of Industry 4.0. In *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)* (pp. 599-604). IEEE. https://doi.org/10.1109/ICNSC.2017.8000159

Braun, R., and Esswein, W. (2014). Classification of Domain-Specific BPMN Extensions. In *The 7th IFIP Working Conference on the Practice of Enterprise Modeling* (pp. 42-57). Springer. https://doi.org/10.1007/978-3-662-45501-2_4

Braun, R., Schlieter, H., Burwitz, M. and Esswein, W. (2016). BPMN4CP Revised -- Extending BPMN for Multi-perspective Modeling of Clinical Pathways. In *2016 49th Hawaii International Conference on System Sciences (HICSS)* (pp. 3249-3258). IEEE. https://doi.ieeecomputersociety.org/10.1109/HICSS.2016.407

Brenner, B., Weippl, E., and Ekelhart, A. (2019). Security Related Technical Debt in the Cyber-Physical Production Systems Engineering Process. In *45th Annual Conference of the IEEE Industrial Electronics Society (IECON)* (pp. 3012-3017). IEEE. https://doi.org/10.1109/IECON.2019.8926646

Brings, J., Daun, M., Bandyszak, T., Stricker, V., Weyer, T., Mirzaei, E., Neumann, M., and Zernickel, J. S. (2019). Model-based documentation of dynamicity constraints for collaborative cyber-physical system architectures: Findings from an industrial case study. *Journal of Systems Architecture, 97*, pp. 153-167. Elsevier. https://doi.org/10.1016/j.sysarc.2019.02.012

Brown, N., Ozkaya, I., Sangwan, R., Seaman, C., Sullivan, K., and Zazworka, N. et al. (2010). Managing technical debt in software-reliant systems. In *The FSE/SDP workshop on Future of software engineering research* (pp. 47-52). New York, NY, USA: ACM. https://doi.org/10.1145/1882362.1882373

Bougouffa, S., Dong, Q. H., Diehm, S., Gemein, F., and Vogel-Heuser, B. (2018). Technical Debt indication in PLC Code for automated Production Systems: Introducing a Domain Specific Static Code Analysis Tool. *IFAC-PapersOnLine: 3rd IFAC*

Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT), *51*(10), pp. 70-75. Elsevier. https://doi.org/10.1016/j.ifacol.2018.06.239

Capitán, L., and Vogel-Heuser, B. (2017). Metrics for Software Quality in Automated Production Systems as an Indicator for Technical Debt. In *2017 13th IEEE Conference on Automation Science and Engineering (CASE)* (pp. 709-716). IEEE. https://doi.org/10.1109/COASE.2017.8256186

Cha, S., Dong, Q. H., and Vogel-Heuser, B. (2018). Preventing Technical Debt for Automated Production System Maintenance using Systematic Change Effort Estimation with Considering Contingent Cost. In *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)* (pp. 595-601). IEEE. https://doi.org/10.1109/INDIN.2018.8472004

Cha, S., Vogel-Heuser, B., and Fischer, J. (2020). Analysis of metamodels for model-based production automation system engineering. *IET Collaborative Intelligent Manufacturing*, *2*(2), pp. 45-55. Wiley. https://doi.org/10.1049/iet-cim.2020.0013

Chergui, M.E.A. and Benslimane, S.M. (2018). A valid BPMN extension for supporting security requirements based on cyber security ontology. In *2018 International Conference on Model and Data Engineering (MEDI)* (pp. 219-232). Springer. https://doi.org/10.1007/978-3-030-00856-7_14

Codabux, Z., and Williams, B. (2013). Managing Technical Debt: An Industrial Case Study. In *2013 4th International Workshop on Managing Technical Debt (MTD)* (pp. 8-15). IEEE. https://doi.org/10.1109/MTD.2013.6608672

Cruz Salazar, L. A., Ryashentseva, D., Lüder, A., and Vogel-Heuser, B. (2019). Cyber-physical production systems architecture based on multi-agent's design pattern—comparison of selected approaches mapping four agent patterns. *The International Journal of Advanced Manufacturing Technology*, *105*(9), pp. 4005-4034. Springer. https://doi.org/10.1007/s00170-019-03800-4

Cunningham, W. (1993). The WyCash Portfolio Management System. *ACM SIGPLAN OOPS Messenger*, *4*(2), pp. 29-30. https://doi.org/10.1145/157710.157715

Davies, R. (2020). *A New Theory for Improving Stakeholder Value from Strategic Change Programmes*. PhD thesis. University of Bristol. Available at: https://research-information.bris.ac.uk/ws/portalfiles/portal/271901780/Final_Copy_2021_03_23_Davies_R_H_PhD.pdf (Accessed: 20 October 2022).

Digkas, G., Ampatzoglou, A., Chatzigeorgiou, A., Avgeriou, P., Matei, O., and Heb, R. (2021). The Risk of Generating Technical Debt Interest: A Case Study. *SN Computer Science*, *2*(1), 12. Springer. https://doi.org/10.1007/s42979-020-00406-6

Dong, Q. H., and Vogel-Heuser, B. (2018). Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey. *IFAC-PapersOnLine: 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, *51*(11), pp. 1192-1199. Elsevier. https://doi.org/10.1016/j.ifacol.2018.08.428

Dong, Q. H., Ocker, F., and Vogel-Heuser, B. (2019). Technical Debt as indicator for weaknesses in engineering of automated production systems. *Production Engineering, 13*(3), pp. 273-282. Springer. https://doi.org/10.1007/s11740-019-00897-0

Dong, Q. H., and Vogel-Heuser, B. (2021a). Modelling technical compromises in electronics manufacturing with BPMN+TD – an industrial use case. *IFAC-PapersOnLine: 17th IFAC Symposium on Information Control Problems in Manufacturing (INCOM), 54*(1), pp. 912-917. Springer. https://doi.org/10.1016/j.ifacol.2021.08.108

Dong, Q. H., and Vogel-Heuser, B. (2021b). Modelling Industrial Technical Compromises in Production Systems with Causal Loop Diagrams. *IFAC-PapersOnLine: 4th IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT), 54*(4), pp. 212-219. Springer. https://doi.org/10.1016/j.ifacol.2021.10.036

Dong, Q. H., and Vogel-Heuser, B. (2022). Including validation of process control systems' engineering into the Technical Debt classification. *Forschung im Ingenieurwesen/Engineering Research.* Springer. *(submitted on 18 March 2022)*

Dong, Q. H., Vogel-Heuser, B., and Neumann, E.-M. (2023). Semi-automatic assessment of lack of control code documentation in automated production systems. *at - Automatisierungstechnik.* De Gruyter. https://doi.org/10.1515/auto-2022-0146 *(accepted on 04 May 2023)*

Dong, Q. H., and Vogel-Heuser, B. (2023). *Prototypical tool BPMN+TD*. Available at: https://github.com/quanghuandong/bpmn_td (Accessed: 04 January 2023).

Fairley, R. E., and Willshire, M. J. (2017). Better Now Than Later: Managing Technical Debt in Systems Development. *Computer, 50*(5), pp. 80-87. IEEE https://doi.org/10.1109/MC.2017.124

Fay, A., Vogel-Heuser, B., Frank, T., Eckert, K., Hadlich, T., and Diedrich, C. (2015). Enhancing a model-based engineering approach for distributed manufacturing automation systems with characteristics and design patterns. *Journal Of Systems And Software, 101*, pp. 221-235. Elsevier. https://doi.org/10.1016/j.jss.2014.12.028

Feldmann, S., Rösch, S., Legat C., and Vogel-Heuser, B. (2014). Keeping requirements and test cases consistent: Towards an ontology-based approach. In *2014 12th IEEE International Conference on Industrial Informatics (INDIN)* (pp. 726-732). IEEE. https://doi.org/10.1109/INDIN.2014.6945603

Fischer, J., Vogel-Heuser, B., Schneider, H., Langer, N., Felger, M., and Bengel, M. (2021). Measuring the Overall Complexity of Graphical and Textual IEC 61131-3 Control Software. *IEEE Robotics And Automation Letters, 6*(3), pp. 5784-5791. IEEE. https://doi.org/10.1109/lra.2021.3084886

Franco, E. F. (2020). *A Dynamical Evaluation Framework for Technical Debt Management in Software Maintenance Process*. PhD thesis. Universidade de São Paulo and the Università degli Studi di Roma “La Sapienza”. Available at:

https://iris.uniroma1.it/retrieve/e3835326-e19b-15e8-e053-a505fe0a3de9/Tesi_dottorato_FerreiraFranco.pdf (Accessed: 20 October 2022).

Fuchs, J., Feldmann, S., Legat, C., and Vogel-Heuser, B. (2014). Identification of Design Patterns for IEC 61131-3 in Machine and Plant Manufacturing. *IFAC Proceedings Volumes*: *19th World Congress of the International Federation of Automatic Control (IFAC)*, *47*(3), pp. 6092-6097. Elsevier. https://doi.org/10.3182/20140824-6-ZA-1003.01595

Graja, I., Kallel, S., Guermouche, N. and Kacem, A.H. (2016). BPMN4CPS: a BPMN extension for modeling cyber-physical systems. In *2016 IEEE 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE)* (pp. 152-157). IEEE. https://doi.org/10.1109/WETICE.2016.41

Guo, Y., Seaman, C., Gomes, R., Cavalcanti, A., Tonin, G., Da Silva, F. Q. B., Santos, A. L. M., and Siebra, C. (2011). Tracking Technical Debt - An Exploratory Case Study. In *2011 27th IEEE International Conference on Software Maintenance (ICSM)* (pp. 528-531). IEEE. https://doi.org/10.1109/ICSM.2011.6080824

Haraldsson, H. V. (2004). *Introduction to system thinking and causal loop diagrams*. Lund, Sweden: Department of chemical engineering, Lund University.

Hristov, D., Hummel, O., Huq, M., and Janjic, W. (2012). Structuring software reusability metrics for component-based software development. In *The 7th International Conference on Software Engineering Advances* (*ICSEA)* (pp. 421-429). Xpert Publishing Services.

Holvitie, J., and Leppanen, V. (2015). Examining Technical Debt Accumulation in Software Implementations. *International Journal of Software Engineering and its Applications*, *9*(6), pp. 109-124. Science and Engineering Research Support Society. https://doi.org/10.14257/ijseia.2015.9.6.12

Holvitie, J., Licorish, S. A., Martini, A., Leppänen, V. (2016). Co-Existence of the 'Technical Debt' and 'Software Legacy' Concepts. In *The 1st International Workshop on Technical Debt Analytics (TDA)* (pp. 80-83). CEUR-WS. Available at: http://ceur-ws.org/Vol-1771/paper14.pdf (Accessed: 20 October 2022).

IEC (2013). *IEC 61131 Programmable Controllers - Part 3: Programming Languages (Third Edition)*. International Electrotechnical Commission (IEC).

Izurieta, C., Ozkaya, I., Seaman, C., Kruchten, P., Nord, R., Snipes, W., and Avgeriou, P. (2016). Perspectives on managing technical debt: A transition point and roadmap from Dagstuhl. In *The 1st International Workshop on Technical Debt Analytics (TDA)* (pp. 84-87). CEUR-WS. Available at: http://ceur-ws.org/Vol-1771/paper15.pdf (Accessed: 20 October 2022).

Izurieta, C., Ozkaya, I., Seaman, C., and Snipes, W. (2017). Technical Debt: A Research Roadmap. *ACM SIGSOFT Software Engineering Notes*, *42*(1), pp. 28-31. New York, NY, USA: ACM. https://doi.org/10.1145/3041765.3041774

ISPE (2022). *GAMP 5 Guide: Compliant GxP Computerized Systems*. International Society for Pharmaceutical Engineering (ISPE). Available at: https://ispe.org/publications/guidance-documents/gamp-5 (Accessed: 20 October 2022).

Klinger, T., Tarr, P., Wagstrom, P., and Williams, C. (2011). An Enterprise Perspective on Technical Debt. In *2011 2nd International Workshop on Managing Technical Debt (MTD)* (pp. 35-38). New York, NY, USA: ACM. https://doi.org/10.1145/1985362.1985371

Kruchten, P., Nord, R. L., and Ozkaya, I. (2012). Technical Debt : From Metaphor to Theory and Practice. *IEEE Software*, *29*(6), pp. 18-21. IEEE. https://doi.org/10.1109/MS.2012.167

Kohn, A., Reif, J., Wolfenstetter, T., Kernschmidt, K., Goswami, S., Krcmar, H., Brodbeck, F., Vogel-Heuser, B., Lindemann, U., and Maurer, M. (2013). Improving Common Model Understanding Within Collaborative Engineering Design Research Projects. In *The 4th International Conference on Research into Design (ICoRD)* (pp. 643-654). Springer. https://doi.org/10.1007/978-81-322-1050-4_51

Lenarduzzi, V., Besker, T., Taibi, D., Martini, A., and Arcelli Fontana, F. (2021). A systematic literature review on Technical Debt prioritization: Strategies, processes, factors, and tools. *Journal of Systems and Software*, *171*, p. 110827. Elsevier. https://doi.org/10.1016/j.jss.2020.110827

Li, Z., Avgeriou, P., and Liang, P. (2015). A systematic mapping study on technical debt and its management. *Journal of Systems and Software*, *101*, pp. 193-220. Elsevier. https://doi.org/10.1016/j.jss.2014.12.027

Li, Z., Liang, P., and Avgeriou, P. (2016). Architecture viewpoints for documenting architectural technical debt. In *Software Quality Assurance* (pp. 85-132). Elsevier. https://doi.org/10.1016/B978-0-12-802301-3.00005-3

Lim, E., Taksande, N., and Seaman, C. (2012). A balancing act: What software practitioners have to say about technical debt. *IEEE Software*, *29*(6), pp. 22-27. IEEE. https://doi.org/10.1109/MS.2012.130

Lüder, A., Klostermeyer, A., Peschke, J., Bratoukhine, A., and Sauter, T. (2005). Distributed Automation: PABADIS versus HMS. *IEEE Transactions on Industrial Informatics*, *1*(1), pp. 31-38. IEEE. https://doi.org/10.1109/TII.2005.843825

Martini, A., Bosch, J., and Chaudron, M. (2015). Investigating Architectural Technical Debt Accumulation and Refactoring over Time: A Multiple-Case Study. *Information and Software Technology*, *67*, pp. 237-253. Elsevier. https://doi.org/10.1016/j.infsof.2015.07.005

Martini, A., and Bosch, J. (2016). Architectural Technical Debt in Embedded Systems. *The INCOSE Proceedings: 26th Annual INCOSE International Symposium*, *26*(1), pp. 1029-1043. Wiley. https://doi.org/10.1002/j.2334-5837.2016.00209.x

Martini, A. and Bosch, J. (2017). On the interest of architectural technical debt: Uncovering the contagious debt phenomenon. *Journal of Software: Evolution and Process*, *29*(10), p. e1877. Wiley. https://doi.org/10.1002/smr.1877

Martini, A., Besker, T., Posch, T., and Bosch, J. (2022). TD Pulse: Assessing the Systematic Management of Technical Debt. *IEEE Software*, pp. 1-8. IEEE. https://doi.org/10.1109/MS.2022.3226035

Melo, A., Fagundes, R., Lenarduzzi, V., and Santos, W. B. (2022). Identification and measurement of Requirements Technical Debt in software development: A systematic literature review. *Journal of Systems and Software*, *194*, p. 111483. Elsevier. https://doi.org/10.1016/j.jss.2022.111483

NAMUR (2022). *NE 185 – Requirements for the qualification of the PCT of modular plants in the GMP environment*. NAMUR – Interessengemeinschaft Automatisierungstechnik der Prozessindustrie e.V.

Neumann, E. M., Vogel-Heuser, B., Fischer, J., Ocker, F., Diehm, S., and Schwarz, M. (2020). Formalization of design patterns and their automatic identification in PLC software for architecture assessment. *IFAC-PapersOnLine: 21st IFAC World Congress*, *53*(2), pp. 7819-7826. Elsevier. https://doi.org/10.1016/j.ifacol.2020.12.1881

Neumann, E. M., Vogel-Heuser, B., Fischer, J., Diehm, S., Schwarz, M., and Englert, T. (2022). Automation software architectures in automated production systems: an industrial case study in the packaging machine industry. *Production Engineering*, *(in press)*. Springer. https://doi.org/10.1007/s11740-022-01133-y

Nugroho, A., Visser, J., and Kuipers, T. (2011). An Empirical Model of Technical Debt and Interest. In *2011 2nd International Workshop on Managing Technical Debt (MTD)* (pp. 1-8). New York, NY, USA: ACM. https://doi.org/10.1145/1985362.1985364

OMG (2011). *Business Process Model and Notation (BPMN), Version 2.0*. Object Management Group (OMG). Available at: http://www.omg.org/spec/BPMN/2.0/PDF/ (Accessed: 20 October 2022).

Ocker, F., Seitz, M., Oligschlager, M., Zou, M., and Vogel-Heuser, B. (2019). Increasing Awareness for Potential Technical Debt in the Engineering of Production Systems. In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)* (pp. 478-484). IEEE. https://doi.org/10.1109/INDIN41052.2019.8972268

Panetto, H., Iung, B., Ivanov, D., Weichhart, G., and Wang, X. (2019). Challenges for the cyber-physical manufacturing enterprises of the future. *Annual Reviews in Control*, *47*, pp. 200-213. Elsevier. https://doi.org/10.1016/j.arcontrol.2019.02.002

Rabiser, R., and Zoitl, A. (2021). Towards Mastering Variability in Software-Intensive Cyber-Physical Production Systems. In *Procedia Computer Science: International Conference on Industry 4.0 and Smart Manufacturing*, *180*, pp. 50-59. Elsevier. https://doi.org/10.1016/j.procs.2021.01.128

Ramasubbu, N., and Kemerer, C. F. (2014). Managing Technical Debt in Enterprise Software Packages. *IEEE Transactions on Software Engineering*, *40*(8), pp. 758-772. IEEE. https://doi.org/10.1109/TSE.2014.2327027

Rehan, R., Knight, M. A., Unger, A. J. A., and Haas, C. T. (2014). Financially sustainable management strategies for urban wastewater collection infrastructure – development of a system dynamics model. *Tunnelling and Underground Space Technology, 39*, pp. 116-129. Elsevier. https://doi.org/10.1016/j.tust.2012.12.003

Rios, N., Mendonça Neto, M. G. de, and Spínola, R. O. (2018). A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. *Information and Software Technology, 102*, pp. 117-145. Elsevier. https://doi.org/10.1016/j.infsof.2018.05.010

Seaman, C., Guo, Y., Zazworka, N., Shull, F., Izurieta, C., Cai, Y., and Vetrò, A. (2012). Using Technical Debt Data in Decision Making: Potential Decision Approaches. In *2012 3rd International Workshop on Managing Technical Debt (MTD)* (pp. 45-48). IEEE. https://doi.org/10.1109/MTD.2012.6225999

Seitz, M., Gehlhoff, F., Cruz Salazar, L. A., Fay, A., and Vogel-Heuser, B. (2021). Automation platform independent multi-agent system for robust networks of production resources in industry 4.0. *Journal of Intelligent Manufacturing*, *32*(7), pp. 2023-2041. Springer. https://doi.org/10.1007/s10845-021-01759-2

Simantics Team (2022). *Open Source Modelling And Simulating Tool For Simantics*. Simantics System Dynamics. Available at: http://sysdyn.simantics.org/ (Accessed: 20 October 2022).

Snipes, W., Robinson, B., Guo, Y., and Seaman, C. (2012). Defining the Decision Factors for Managing Defects: A Technical Debt Perspective. In *2012 3rd International Workshop on Managing Technical Debt (MTD)* (pp. 54-60). IEEE. https://doi.org/10.1109/MTD.2012.6226001

SonarSource SA (2022). *SonarQube – Code Quality and Code Security*. Available at: https://www.sonarqube.org/ (Accessed: 06 December 2022).

Sterman, J. (2000). *Business dynamics: Systems thinking and modeling for a complex world*. McGraw-Hill/ Irwin.

Stopford, B., Wallace, K., and Allspaw, J. (2017). Technical Debt Challenges and Perspectives. *IEEE Software, 34*(4), pp. 79-81. https://doi.org/10.1109/MS.2017.99

Strauss, A. and Corbin, J. M. (1997). *Grounded theory in practice*. SAGE.

Stroppi, L.J.R., Chiotti, O. and Villarreal, P.D. (2011). Extending BPMN 2.0: method and tool support. In *2011 International Workshop on Business Process Modeling Notation* (pp. 59-73). Springer. https://doi.org/10.1007/978-3-642-25160-3_5

Szabados, K., and Kovacs, A. (2015). Technical Debt of Standardised Test Software. In *2015 IEEE 7th International Workshop on Managing Technical Debt (MTD)* (pp. 57-60). IEEE. https://doi.ieeecomputersociety.org/10.1109/MTD.2015.7332626

Tichy, M., Henkler, S., Holtmann, J., and Oberthür, S. (2008). Component Story Diagrams: A Transformation Language for Component Structures in Mechatronic Systems. In *4th Workshop on Object-Oriented Modeling of Embedded Real-Time Systems*

(OMER) (pp. 27-38). Paderborn, Germany: Heinz-Nixdorf-Institut. https://www.hni.uni-paderborn.de/pub/7036

Tom, E., Aurum, A., and Vidgen, R. (2013). An Exploration of Technical Debt. *Journal Of Systems And Software*, *86*(6), pp. 1498-1516. Elsevier. https://doi.org/10.1016/j.jss.2012.12.052

Trunzer, E. (2020). *T Model-driven System Architectures for Data Collection in Automated Production Systems*. PhD thesis. Technical University of Munich, Germany. Available at: http://mediatum.ub.tum.de/doc/1545842/1545842.pdf (Accessed: 20 October 2022).

Ulewicz, S. (2018). *Test Coverage Assessment for Semi-Automatic System Testing and Regression Testing Support in Production Automation*. PhD thesis. Technical University of Munich, Germany. Available at: https://mediatum.ub.tum.de/doc/1431646/1431646.pdf (Accessed: 20 October 2022).

VDI (2021). *VDI/VDE 2206 Entwicklung mechatronischer und cyber-physischer Systeme*. Berlin, Germany: Beuth Verlag. Available at: https://www.vdi.de/richtlinien/programme-zu-vdi-richtlinien/vdi-2206 (Accessed: 20 October 2022).

Vogel-Heuser, B., Folmer, J., and Legat, C. (2013). Anforderungen an Die Softwareevolution in Der Automatisierung Des Maschinen-Und Anlagenbaus: Requirements on Software Evolution in Machine and Plant Automation. *at - Automatisierungstechnik*, *62*(3), pp. 163-174. De Gruyter. https://doi.org/10.1515/auto-2013-1051

Vogel-Heuser, B., and Rösch, S. (2015a). Applicability of Technical Debt as a Concept to Understand Obstacles for Evolution of Automated Production Systems. In *2015 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 127-132). IEEE. https://doi.org/10.1109/SMC.2015.35

Vogel-Heuser, B., Rösch, S., Martini, A., and Tichy, M. (2015b). Technical Debt in Automated Production Systems. In *2015 IEEE 7th International Workshop on Managing Technical Debt (MTD)* (pp. 49-52). IEEE. https://doi.org/10.1109/MTD.2015.7332624

Vogel-Heuser, B., Fay, A., Schaefer, I., and Tichy, M. (2015c). Evolution of Software in Automated Production Systems: Challenges and Research Directions. *Journal Of Systems And Software*, *110*, pp. 54-84. Elsevier. https://doi.org/10.1016/j.jss.2015.08.026

Vogel-Heuser, B., Fischer, J., Feldmann, S., Ulewicz, S., and Rösch, S. (2017a). Modularity and Architecture of PLC-Based Software for Automated Production Systems: An Analysis in Industrial Companies. *Journal Of Systems And Software*, *131*, pp. 35-62. Elsevier. https://doi.org/10.1016/j.jss.2017.05.051

Vogel-Heuser, B., Heinrich, R., Cha, S., Rostami, K., Ocker, F., Koch, S., Reussner, R. and Ziegltrum, S. (2017b). Maintenance effort estimation with KAMP4aPS for cross-

disciplinary automated PLC-based Production Systems - a collaborative approach. *IFAC-PapersOnLine: 20th IFAC World Congress*, *50*(1), pp. 4360-4367. Elsevier. https://doi.org/10.1016/j.ifacol.2017.08.877

Vogel-Heuser, B., and Neumann, E.-M. (2017c). Adapting the concept of technical debt to software of automated Production Systems focusing on fault handling, mode of operation and safety aspects. *IFAC-PapersOnLine: 20th IFAC World Congress*, *50*(1), pp. 5887-5894. Elsevier. https://doi.org/10.1016/j.ifacol.2017.08.1308

Vogel-Heuser, B., Fischer, J., Neumann, E. M., and Diehm, S. (2018). Key maturity indicators for module libraries for PLC-based control software in the domain of automated Production Systems. *IFAC-PapersOnLine: 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, *51*(11), pp. 1610-1617. Elsevier. https://doi.org/10.1016/j.ifacol.2018.08.261

Vogel-Heuser, B., Brodbeck, F., Kugler, K., Passoth, J., Maasen, S., and Reif, J. (2020a). BPMN+I to support decision making in innovation management for automated production systems including technological, multi team and organisational aspects. *IFAC-PapersOnLine: 21st IFAC World Congress*, *53*(2), pp. 10891-10898. Elsevier. https://doi.org/10.1016/j.ifacol.2020.12.2825

Vogel-Heuser, B., Böhm, M., Brodeck, F., Kugler, K., Maasen, S., Pantförder, D., Zou, M., Buchholz, J., Bauer, H., Brandl, F., and Lindemann, U. (2020b). Interdisciplinary engineering of cyber-physical production systems: highlighting the benefits of a combined interdisciplinary modelling approach on the basis of an industrial case. *Design Science*, *6*, p. e5. Cambridge University Press. https://doi.org/10.1017/dsj.2020.2

Vogel-Heuser, B. and Bi, F. (2021). Interdisciplinary effects of technical debt in companies with mechatronic products — a qualitative study. *Journal of Systems and Software*, *171*, p. 110809. Elsevier. https://doi.org/10.1016/j.jss.2020.110809

Vogel-Heuser, B., Reif, J. A. M., Passoth, J.-H., Huber, C., Brodbeck, F. C., Maasen, S., Lindemann, U., and Hujo, D. (2022a). BPMN++ to support managing organisational, multiteam and systems engineering aspects in cyber physical production systems design and operation. *Design Science*, *8*, p. e4. Cambridge University Press. https://doi.org/10.1017/dsj.2021.29

Vogel-Heuser, B., Neumann, E. M., Fischer, J., Marcos, M., Estevez, E. E., Barbieri, G., Sonnleithner, L., and Rabiser, R. (2022b). Automation Software Architecture in CPPS - Definition, Challenges and Research Potentials. In *2022 IEEE 5th International Conference on Industrial Cyber-Physical Systems (ICPS)* (pp. 01-08). IEEE. https://doi.org/10.1109/ICPS51978.2022.9816893

Vogel-Heuser, B., Neumann, E.-M., and Fischer, J. (2022c). MICOSE4aPS: Industrially Applicable Maturity Metric to Improve Systematic Reuse of Control Software. *ACM Transactions on Software Engineering and Methodology*, *31*(1), pp. 1-24. New York, NY, USA: ACM. https://doi.org/10.1145/3467896

Vyatkin, V. (2013). Software engineering in industrial automation: State-of-the-art review. *IEEE Transactions on Industrial Informatics*, *9*(3), pp. 1234-1249. IEEE. https://doi.org/10.1109/TII.2013.2258165

Waltersdorfer, L., Rinker, F., Kathrein, L., and Biffl, S. (2020). Experiences with technical debt and management strategies in production systems engineering. In *The 3rd International Conference on Technical Debt* (pp. 41-50). New York, NY, USA: ACM. https://doi.org/10.1145/3387906.3388627

Wiklund, K., Eldh, S., Sundmark, D., and Lundqvist, K. (2012). Technical Debt in Test Automation. In *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation* (pp. 887-892). IEEE. https://doi.org/10.1109/ICST.2012.192

Wilch, J., Fischer, J., Langer, N., Felger, M., Bengel, M., and Vogel-Heuser, B. (2022). Towards automatic generation of functionality semantics to improve PLC software modularization. *at - Automatisierungstechnik* , *70*(2), pp. 181-191. De Gruyter. https://doi.org/10.1515/auto-2021-0138

Yousfi, A., Freitas, A.A., Dey, A.K. and Saidi, R. (2015). The use of ubiquitous computing for business process improvement. *IEEE Transactions on Services Computing*, *9*(4), pp. 621-632. IEEE. https://doi.org/10.1109/TSC.2015.2406694

Zarour, K., Benmerzoug, D., Guermouche, N., and Drira, K. (2019). A systematic literature review on BPMN extensions. *Business Process Management Journal*, *26*(6), pp. 1473-1503. Emerald Publishing Limited. https://doi.org/10.1108/BPMJ-01-2019-0040

Zazworka, N., Seaman, C., and Shull, F. (2011). Prioritising Design Debt Investment Opportunities. In *2011 2nd International Workshop on Managing Technical Debt (MTD)* (pp. 39-42). New York, NY, USA: ACM. https://doi.org/10.1145/1985362.1985372

Zazworka, N., Vetro', A., Izurieta, C., Wong, S., Cai, Y., Seaman, C., and Shull, F. (2014). Comparing four approaches for technical debt identification. *Software Quality Journal*, *22*(3), pp. 403-426. Springer. https://doi.org/10.1007/s11219-013-9200-8

Zimmermann, M., Königs, S., Niemeyer, C., Fender, J., Zeherbauer, C., Vitale, R., and Wahle, M. (2017). On the design of large systems subject to uncertainty. *Journal of Engineering Design*, *28*(4), pp. 233-254. Taylor & Francis. https://doi.org/10.1080/09544828.2017.1303664

7. Table of Figures

8. Table of Tables

9. Appendix – included papers

This part consists of six included papers.

[***Paper1***] Dong, Q. H., and Vogel-Heuser, B. (2018). Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey. *IFAC-PapersOnLine: 16th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, *51*(11), pp. 1192-1199. Elsevier. https://doi.org/10.1016/j.ifacol.2018.08.428

[***Paper2***] Dong, Q. H., Ocker, F., and Vogel-Heuser, B. (2019). Technical Debt as indicator for weaknesses in engineering of automated production systems. *Production Engineering*, *13*(3), pp. 273-282. Springer. https://doi.org/10.1007/s11740-019-00897-0

[***Paper3***] Dong, Q. H., and Vogel-Heuser, B. (2021a). Modelling technical compromises in electronics manufacturing with BPMN+TD – an industrial use case. *IFAC-PapersOnLine: 17th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)*, *54*(1), pp. 912-917. Springer. https://doi.org/10.1016/j.ifacol.2021.08.108

[***Paper4***] Dong, Q. H., and Vogel-Heuser, B. (2021b). Modelling Industrial Technical Compromises in Production Systems with Causal Loop Diagrams. *IFAC-PapersOnLine: 4th IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control (CESCIT)*, *54*(4), pp. 212-219. Springer. https://doi.org/10.1016/j.ifacol.2021.10.036

[***Paper5***] Dong, Q. H., and Vogel-Heuser, B. (2022). Including validation of process control systems' engineering into the Technical Debt classification. *Forschung im Ingenieurwesen/Engineering Research.* Springer. *(submitted on 18 March 2022)*

[***Paper6***] Dong, Q. H., Vogel-Heuser, B., and Neumann, E.-M. (2023). Semi-automatic assessment of lack of control code documentation in automated production systems. *at - Automatisierungstechnik.* De Gruyter. https://doi.org/10.1515/auto-2022-0146 *(accepted on 04 May 2023)*

Paper1

IFAC-PapersOnLine 51/11 (2018), pp. 1192-1199.
https://doi.org/10.1016/j.ifacol.2018.08.428

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 51-11 (2018) 1192–1199

Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey

Quang Huan Dong* Birgit Vogel-Heuser*

** Technical University of Munich, Institute of Automation and Information Systems, 85748 Garching near Munich, Germany (e-mail: {huan.dong, vogel-heuser}@tum.de).*

Abstract: Technical Debt (TD) metaphor was introduced at classical software engineering domain more than two decades ago. Outcomes of TD research have been gradually showing the benefits, which support practitioners to identify and remove TD items, thus reduce the maintenance cost. Recently, TD concept has just been successfully introduced into automated Production Systems (aPS) domain. aPS have long operation time, which has significant maintenance cost. In aPS domain, the research about TD is more complex, because besides software, aPS consist of other disciplines such as mechanical and electrical parts. Strong relationships of these three disciplines have substantial influences to development and maintenance phases of aPS. Thus, it is a challenge to study TD items in this domain. We need a broader investigation with involvements from multiple departments, to examine TD in aPS, rather than just analyze software part. However, up to now, most researches about TD in this domain still focus on software part only. This research studies TD items in aPS concerning the surround factors and impacts of these factors to the software side. The research method is TD4aPS, an adaption of SWMAT4aPS benchmark process to TD research for aPS. We select and present two typical case studies in the paper. One case is from a machine manufacturer. Another case is from a plant manufacturer. Two typical characteristics of TD in aPS domain are shown: cross-disciplinary and cross-life-cycle-phase. From analysis, the detected TD items are classified then short- and long-term recommendations are proposed. We validated qualitative results in case studies by quantitative results from an online survey received from German industrial companies working with aPS at 40 different markets.

Keywords: Integration of Knowledge/Competence in Enterprise Modelling Framework, Enterprise System Quality Improvement, PLM Systems

1. INTRODUCTION

Coined by Cunningham (1993), Technical Debt (TD) is a metaphor used to describe the long-term negative impact from a sub-optimal solution in the source code to meet the deadline. TD causes significant effort to maintain a software system. The TD concept has been significantly paid attention and up to now, it is a well-known research topic to both academia and industry in classical software engineering domain, according to Li et al. (2015). TD is a friction to software development, see Avgeriou et al. (2016a). However, TD is still unfamiliar in aPS domain. In aPS domain, the production process is driven by the software running in the programmable logic controllers (PLC). The development process of aPS is more complex than development process of classical software engineering process. aPS is developed in a jointly process, which three sub-processes occur in parallel at the mechanical, electrical and software disciplines. The processes at three disciplines are not started at the same time but slightly overlapped, because of the dependencies. The mechanical design phase usually starts first. When mechanical engineers have a preliminary design, process at electrical engineering is started. The electricians are responsible for the connection between sensors, actuators and PLC. In consequence, software engineers are responsible for developing PLC programs. aPS have long life time up to decades. There are machine and plant manufacturers in aPS domain. While the machine manufacturers could integrate and test the systems before a delivery, the plant manufacturers could not be able to verify the whole systems due to heaviness and large dimensions of the modules. Thus, the first time that many systems are fully integrated and verified, is often at commissioning phase, which happens at a customer site. During this phase, on-site engineers are always under a high pressure, because the customer wants to start the production at the earliest. This pressure could be a trigger of choosing sub-optimal solution if an issue occurs during commissioning period. Recently, applicability of TD concept into aPS domain was successfully demonstrated in Vogel-Heuser et al. (2015a). An open research direction is to collect more examples of TD items from industrial companies to build a comprehensive TD classification for aPS domain (Vogel-Heuser et al., 2015b).

In this research, besides three mentioned disciplines (i.e. mechanical, electrical and software parts), the studied systems also include hydraulic or pneumatic sides. Hereafter, the term *mechanics* includes mechanical, hydraulic and pneumatic aspects. Based on the classification in Li et al. (2015), the main contributions of this research are to: ① collect and analyze more examples of TD items from case studies in industrial aPS companies; ② propose short- and

2405-8963
Peer review under responsibility of International Federation of Automatic Control.
10.1016/j.ifacol.2018.08.428

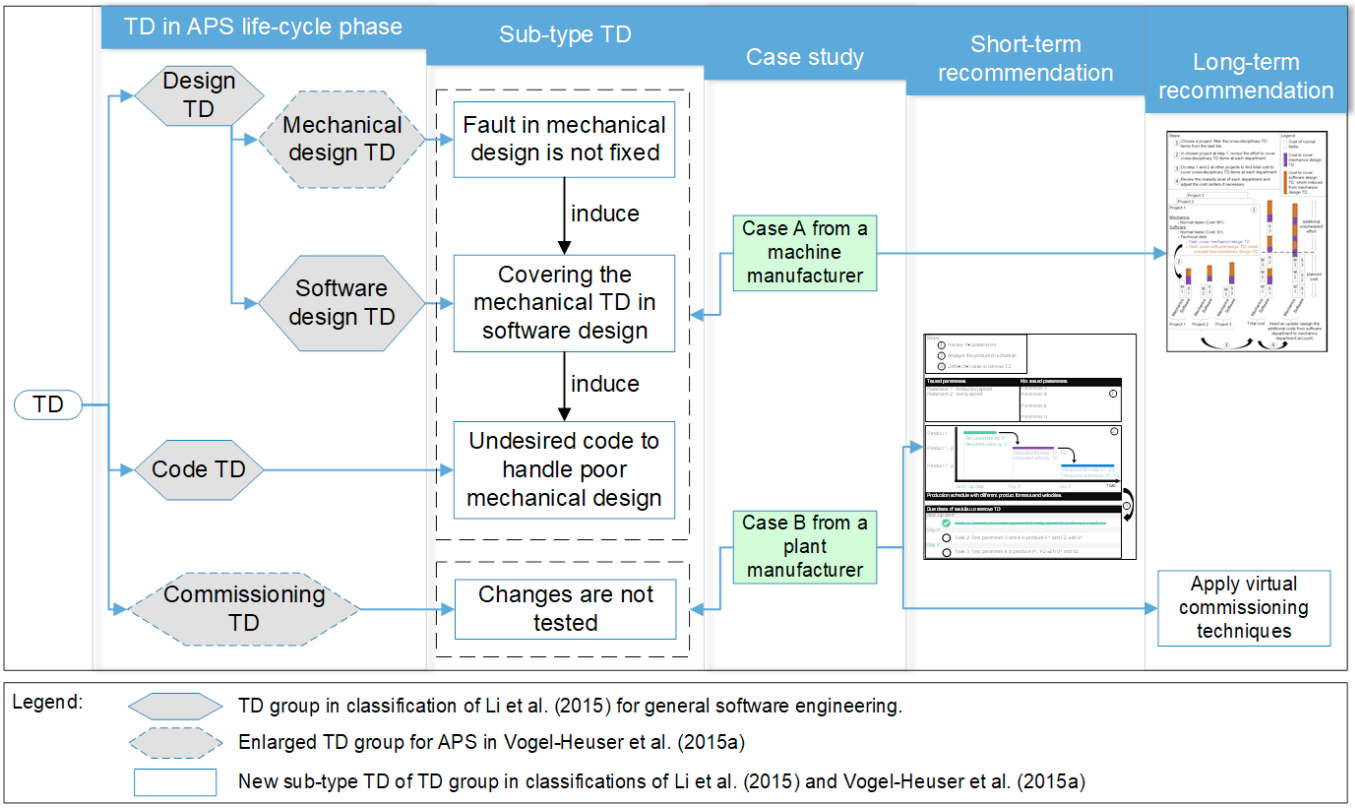


Fig. 1. Taxonomy of indicated TD items from case studies at production systems in this research

long-term recommendations for identified TD items; ③ extend current TD classification with cross-disciplinary, cross-life-cycle-phase sub-type TD (see Fig. 1). Another contribution is TD4aPS - Technical Debt for aPS - process. We adapt SWMAT4aPS benchmark process (Vogel-Heuser et al., 2017a) to TD4aPS as a research method for TD research in aPS domain.

2. RELATED WORK IN RESEARCHING TD IN SOFTWARE ENGINEERING AND aPS DOMAINS

There have been many studies about causes of TD and TD impacts on maintenance cost in classical software engineering domain. Seaman et al. (2015) indicated many potential avenues of further TD research at other communities and domains. However, TD studies in aPS domain are still limited so far.

In classical software engineering domain, a comprehensive classification of TD types and subtypes was presented by Li et al. (2015). Avgeriou et al. (2016b) indicated TD always relates to cost. First, TD is the effort, which was saved by developing a low quality software to meet the product delivery date. After that, TD is the cost, which has to be spent in the refactoring activities for software system. Furthermore, violations on architecture level (e.g., improper software design) might indicate maintenance problems, as Zazworka et al. (2014) pointed out.

Near to aPS domain, Martini et al. (2016) successfully identified some factors causing TD in embedded software development. One significant factor triggering TD is the pressure to deliver the systems on time. This study showed that if TD is not managed, then a large refactoring or a redesign is needed in order to add new features into the system later.

In aPS domain, Besker et al. (2017) conducted a study at one company developing aPS. This study identified TD was causing significant additional effort as the company "*spends quite a lot of resources in paying the interest on Technical Debt, on average 32 % of the development time*" but TD was not recognized at a company level yet. Focus of this research was on software discipline and the authors indicated the importance to conduct further study on how different disciplines interact with each other in a TD perspective.

Vogel-Heuser et al. (2017b) focused on reusability and architecture of aPS software at several industrial companies. This study proposed some guidelines to prevent similar TD items indicated in code structure of aPS software. Capitán et al. (2017) tried to adapt some existing software metrics from the software engineering domain to aPS domain. The research focused on TD items regarding size, complexity and modularity aspects of aPS software. This study was performed on a laboratory aPS demonstrator. Two above studies found many new TD subtypes, according to classification of Li et al. (2015). However, focuses of these studies were still at software discipline of aPS.

Vogel-Heuser et al. (2015b) reported some examples of TD items at other disciplines (i.e. mechanical and electrical) of aPS. Vogel-Heuser et al. (2015a) enlarged the TD classification of Li et al. (2015) for aPS domain. Enlarged TD types described in these two studies were based on additional

aPS life cycle phases in compared to software life cycle phases of classical software engineering domain (see Fig. 3 for aPS life cycle phases). However, the presented TD items were from personal experience of the authors or from a laboratory demonstrator only.

In conclusion, to the best of our knowledge, there is lacking a systematic study about TD items causing cross-disciplinary impact in industrial companies in aPS domain. This research aim is to fulfil this gap by conducting case studies at industrial aPS companies with involvements from multiple disciplines. Cross-life-cycle-phase TD items would also be examined. Thus, our contribution would gain a broader knowledge about TD in this domain.

3. STEPS OF THE RESEARCH

This section describes the activities conducted with participated machine and plant manufacturers in Germany (target groups of the study). We revised the benchmark process for aPS software maturity (SWMAT4aPS) in Vogel-Heuser et al. (2017a) to TD4aPS to serve new focus of this research: cross-disciplinary and cross-life-cycle-phase TD in aPS. The adapted process is illustrated in Fig. 2.

At preparation phase, we retrieved necessary data about the existing obstacles in aPS development then developed two questionnaires by some initial interviews (activity ①). The essay questionnaire was intended for interviews at next phases. The multiple choice questionnaire was intended for an online survey. The multiple choice questionnaire was an enhancement of an existing survey (Vogel-Heuser et al., 2017a) focusing on software maturity. Activity ① was a revision of the corresponding activity in SWMAT4aPS in order to adapt to new context in this research.

A new phase (beta test) was added because of two reasons. First, we wanted to check the required time which the questionnaires could be finished, because we assumed industrial people could only spend short time for our activities. Second, we wanted to check if the multiple choice questionnaire could gain a holistic view on current development and maintenance activities in the company of a participant (in online survey) and the essay questionnaire could gain enough information of a case (at interviews) for further analysis. First, based on the essay questionnaire, interviews were executed with two experts from industry in activity ②, who had some knowledge about TD already. These experts were a leader from development department and an engineer from quality department. Two experts were from different companies. The received feedback from experts was incorporated into both questionnaires. For example, *technical debt* term should not be directly introduced when starting an interview. It might be misunderstood as a development process, if an interviewee is new to TD concept. It should be better to use *workaround* term, which is more familiar to people in industry, when starting a discussion then TD definition could be consequently introduced. Another feedback was about reasons of suboptimal solutions: "*Rushed to meet the delivery date*" and "*Pressure to keep the system in operation*" should be generalized to "*Time pressure*". Second, in activity ③, we gained assessments from two other industrial experts to enhance the efficiencies of the multiple choice questionnaire. We selected candidates from different companies to the companies in activity ② to gain suggestions

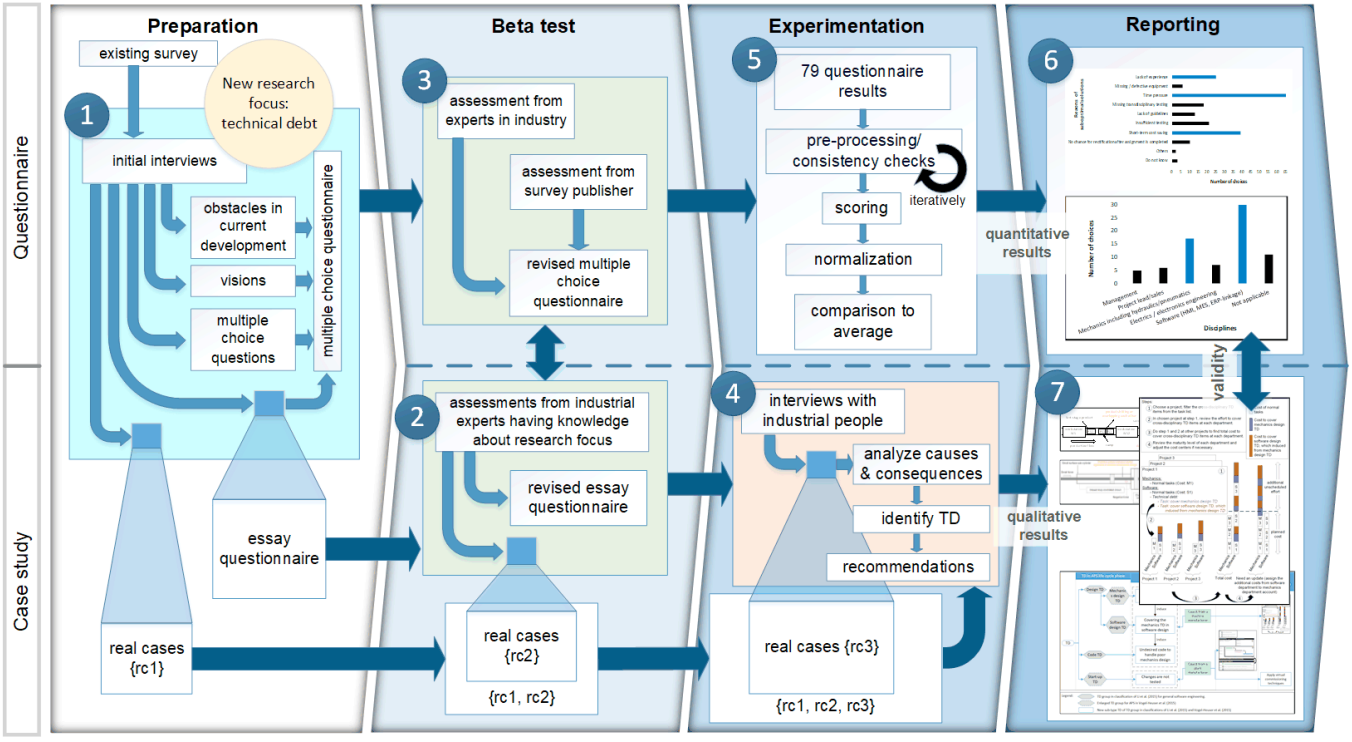


Fig. 2. TD4aPS - an adaption of SWMAT4aPS (Vogel-Heuser et al., 2017a) for TD research in aPS domain (1: revised activity due to new research focus; 2, 3: new activities; 4: replacement of expert analysis activity; 5, 6, 7: existing activities)

from different contexts. An example of valuable feedback to us was: *"We should ask for the number of workarounds being longer than 1 year, 3 years or 5 years in the system. The aim of that questions is to indicate whether there arises an accumulation of TD and whether there is a critical value that indicates a high increase of effort in order to dissolve the workarounds (e.g. workarounds building up on each other, new development based on the workaround)."* We also retrieved assessments of people from our survey publisher. Instead of broadcasting the survey by ourselves, we chose a professional publishing service, which could reach large number of participants in our target groups to ensure quality of the responses. Feedback from people at the publisher also supported us to order, show or hide the questions properly, according to the choices of previous questions. As a result, although we added many questions into the existing questionnaire (the revised version has more than 50 questions), we were still able to maintain the minimum required time, in which a participant needs to answer all questions, approximately 20 minutes in average (the upper limit for an online survey, according to publishing experts' experience).

Two final phases (experimentation and reporting) were similar to corresponding phases in SWMAT4aPS. There was just a minor adjustment which activity ④ was a replacement of expert analysis activity. Other activities (⑤, ⑥ and ⑦) were existing activities in SWMAT4aPS. Activity ④ and ⑤ were performed in parallel. First, the main interviews were conducted at other industrial organizations based on the revised essay questionnaire (activity ④). These interviewees were with people from multiple disciplines (i.e. electronic engineering, control systems engineering, instrumentation engineering and mechanics engineering). Roles of the informants varied from manager to engineer. During this period, we found two interesting cases (reported at next section, Section 4). More investigations (e.g. additional questions, interactions, etc.) were performed with the interviewees to study these two cases carefully. The analysis was not only based on data of current cases, which reported by the interviewees in this activity {rc3}, but also referred to obtained data at previous activities {rc1, rc2} as a source of information for similar cases. Hence, discussions as well as issue detection could be performed faster after each interview because participants could only spend limited time (one hour in average) for each the interview as aforementioned. Second, we illustrated the qualitative results from activity ④ in activity ⑦. We validated these results by quantitative results from an online survey from activity ⑤ and ⑥. Data from the online survey was processed in activity ⑤ then illustrated in activity ⑥. Finally, reporting activities aimed to provide a holistic view about the participated aPS manufactures. On one hand, the quantitative results of activity ⑥ focused on whole status of development and maintenance, thus, could be used to validate the results in ⑦. On the other hand, the qualitative results of activity ⑦ could provide in detail cases which TD items causing unscheduled efforts to participated aPS manufacturers.

4. ENLARGEMENT OF TD CLASSIFICATION FROM CASE STUDIES AND CORRESPONDING RECOMMENDATIONS

In this section, two typical cases studies are selected and described from a machine manufacturer in case A and from a plant manufacturer in case B. The issues in case studies are analyzed then indicated TD items are classified. The TD in case A occurs at the detailed design phase of an aPS. The TD in case B arises at commissioning phase, after aPS are developed and delivered to customer site. The starting points of indicated TD items referring to aPS life cycle are briefly presented in Fig. 3. After an analysis to detect TD items, short- and long-term recommendations are proposed to each specific case. TD items in these case studies show typical characteristics of TD in aPS domain: cross-disciplinary TD (in case A) and cross-life-cycle-phase TD (in case B). Due to confidentiality agreements with the participated companies, only abstracted results are available.

4.1 Case A – machine manufacturer – Design TD introduced by mechanical engineers

In one product line of this manufacturer, the machines are equipped with multiple-stage hydraulic cylinders, which are used to generate forces. A multiple-stage hydraulic cylinder

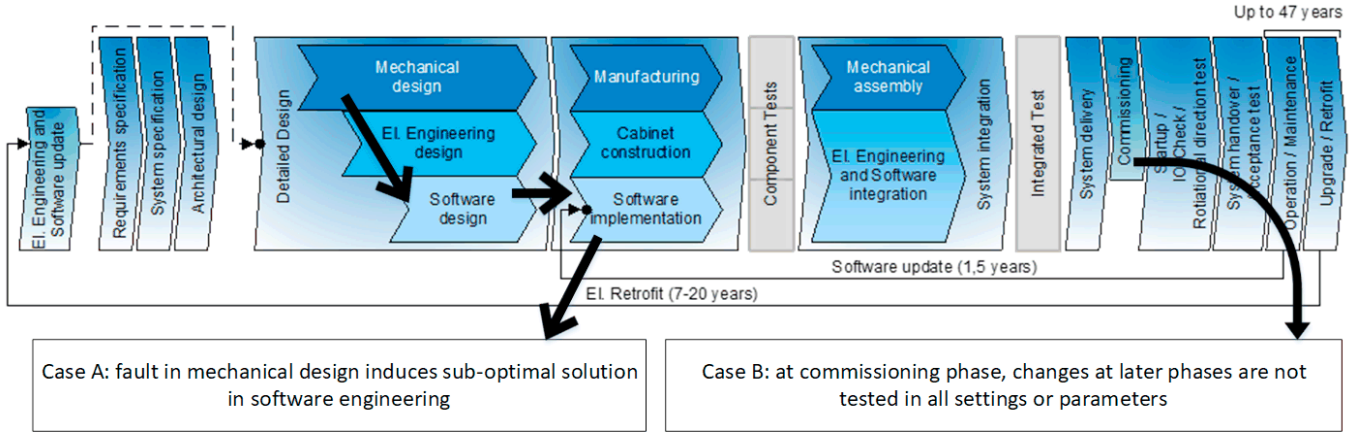


Fig. 3. TD in two cases according to life cycle phases of aPS in Vogel-Heuser et al. (2015b)

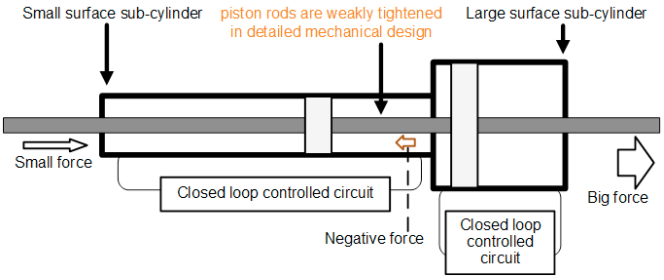


Fig. 4. Potential safety issue to the machines in case A (weak connection could be broken if big negative force occurs)

contains multiple sub-cylinders, which have different surface sizes. The sub-cylinders are ordered by surface sizes (the force is increased from small to large surface sub-cylinders following Pascal's principle). Based on the detailed design, piston rods of these sub-cylinders are mechanically tightened together. Regarding the control system, the cylinders are controlled by closed loop controlled circuits (one for each sub-cylinder). This design has some weak points at connection of piston rods. There could be some negative forces and these forces will stress the mechanical connection (see Fig. 4). If the connection is broken, the cylinders could be damaged. To avoid these negative forces, a workaround in control software was developed observing the forces in the sub- cylinders to adjust control parameters if negative forces occur. These control parameters need to be optimized repeatedly because the cylinder movement is complex. Thus, it will last a concern during the complete lifetime of these machines. According to the interviewee, the hydraulic cylinders need to be re-designed. However, due to technical and cost reasons, a re-design activity for these cylinders was never scheduled because the development project of these machines was closed commercially already. Impact of the issue in the mechanical part was rated as high and in the control software part as medium level.

4.2 Case B – plant manufacturer – commissioning phase

This manufacturer builds plants for furniture and building companies. In these aPS, at a workstation (WS1), raw materials are manipulated to form the first stage of a product (PS1). The PS1 products are then transported on a ramp to next workstation (WS2) in the production process. To optimize energy consumption, at WS2, the products are operated in batch, which means several PS1 items from the line are collected before the procedure at WS2 is triggered, constructing the second stage of the products (PS2 items). The process can be configured by control software settings or parameters so called process settings. This plant manufacturer is expanding its market boundary across continents and the needs of new customers might be different to current capacity of the systems. During commissioning phase at a customer's site, an on-site engineer identified reason for the lack of functionality and adjusted production- and ramp-speed parameters by using a trial and error strategy. Adjusting these two parameters only worked for certain range of product formats (e.g. product dimensions). When the engineer tried producing other ranges in order to adapt with changes (i.e. different raw materials, different velocities) at later phases, items on the ramp were hitting or overlapping each other when being collected prior to WS2. The hitting/overlapping issue resulted in damaged PS2 products afterwards (illustrated in Fig. 5). However, the engineer abandoned to perform any further experiments or adjustments for other formats immediately because of time pressure to produce first products to be sold. This could be an intentional easier decision for the engineer and less trouble with the customer. Thus, he took TD, which will cost even more money in the later phases. The issue in control software parameters was ranked as high level.

4.3 TD analysis and classification

As the development process of aPS involves multiple disciplines, when a sub-optimal solution from one discipline causes negative affects to other discipline(s), it is considered as a cross-disciplinary TD. When interest of TD spreads to different aPS life cycle phase(s), it is considered as a cross-life-cycle-phase TD. During the analysis, when a new TD item is indicated, we compared its type with the existing TD categories in the classifications of Li et al. (2015) and Vogel-Heuser et al. (2015a). The taxonomy of TD items identified from this research is presented in Fig. 1.

In case A, the first TD item (fault in mechanical design is not fixed) is classified into *Mechanical design TD* group. This mechanical design fault induces a software design TD (covering the mechanic TD in software design) introducing additional code, which does not fit into the control code architecture or any library module to be used for future plants. The cylinder movement is over controlled in software. An undesired extension of control software functionality has to be realized due to constraints from mechanical discipline. Thus, this TD in mechanical design is a cross-disciplinary TD because interest of this TD has extended to the software discipline. This modularity violation causes low quality of the system such as low decomposability. As aforementioned, violation in modularity might indicate maintenance problems. For example, a variant is generated which will only be used once but needs to be maintained for all software updates of this plant. Another example is, it is difficult to isolate the modules for investigation of another kind of issues.

In case B, the intended delay in testing changes at later phases in all settings or parameters resulted in a TD. The cost to test process settings for those changes could be foreseen as an addition cost of next life cycle phases, when the plant runs more efficiently with higher quality and throughput.

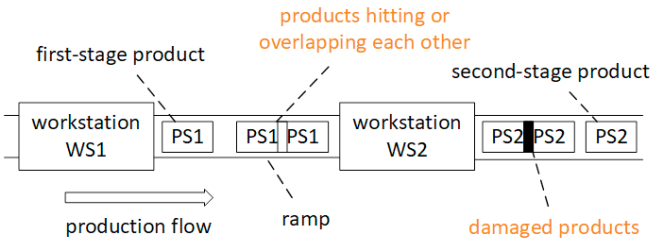


Fig. 5. Production issue in case B due to insufficient configurations in process settings

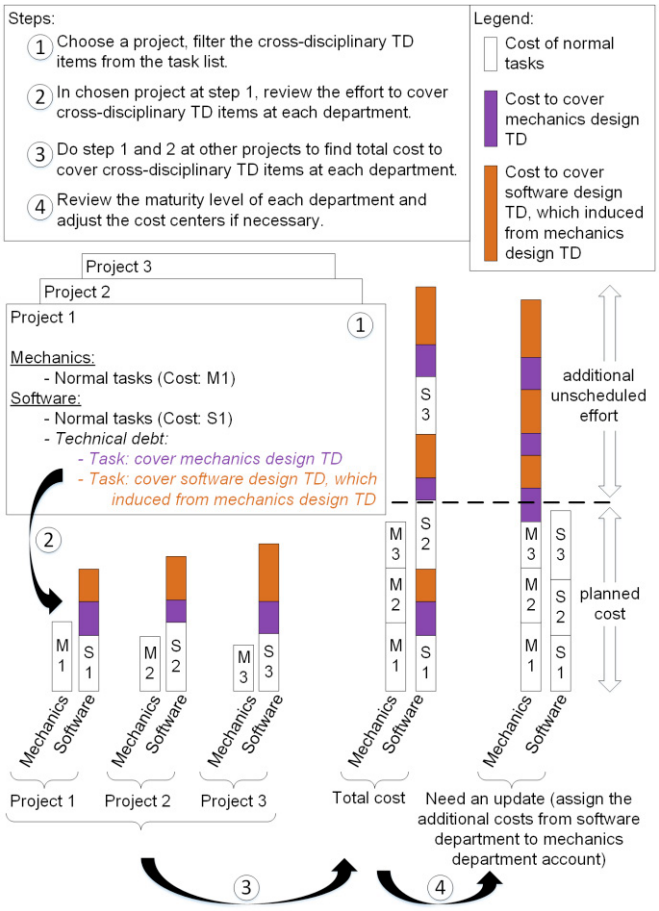


Fig. 6. Long-term recommendation for case A

Moreover, the production of customer would need to be suspended while the engineer is testing process settings for each new situation. Hence, long-term negative impact from this TD becomes apparent to next life cycle phases due to this TD at commissioning phase. This cross-life-cycle-phase TD is *Changes are not tested.* Since it is at the commissioning phase already, this TD would not belong to the test phase. Thus, this TD is categorized into *Commissioning TD* group.

4.4 Recommendations to prevent/remove TD

Before providing a recommendation, we tried evaluating surround factors (e.g. from financial and controlling matters) to increase the possibilities which the recommendation can be considered and applied in short- or long-term perspective. From the interviews, an important information, which should be noted, is even if the aPS manufacturers would like revise the faulty design artefacts after a project has been delivered and TD has occurred, they may not be able to because they cannot assign necessary working hours to correct the fault to this old project any more. It is because the project was closed commercially already. This reason is also confirmed by personal experience of the authors when working in industry.

In case A, it is about a closer cooperation in between disciplines. Unfortunately, in this situation there is no other way than accepting cross-disciplinary TD due to time delivery fixed in the contract and the project is already closed. If an additional unscheduled activity was added in a project (e.g. extra effort to develop an extension in software in this case study), there is a high chance that actual cost has exceeded planned cost. In addition, this situation might occur at other projects which might use shared resources (e.g. designers or technical designs) or follow similar development process to the project in this case study. Thus, for this case, a long-term recommendation is to collect different TD items in between different departments gathered over different projects. The company should start to inspect spent time from each discipline in the projects to filter the cost, which is used to cover TD from other disciplines. These steps could support the company to indicate the maturity level of each department to assign the appropriate cost centers. In case it is always the software department which has to cover mechanics TD this would be a good indicator for the management team about the maturity of the mechanics department and it would be necessary to assign the additional costs in the software department to the mechanics department accounts (cost centers). Fig. 6 summarizes the recommendation in four steps (from ① to ④) and illustrates an example result of this case (mechanics discipline has low maturity level) after applying these steps.

In case B, for short-term resolution to remove TD, a possible approach is, the on-site engineer would study the production agenda of the customer then update maintenance task list accordingly. These "outlook" tasks can support the on-site engineer to plan and remember that there are some tasks, which need to be done before a production situation changes. The steps and possible tasks are illustrated in Fig. 7. These tasks need to be checked regularly. Otherwise, there could be another trouble if some components (e.g. the ramp or the sensors) are impacted when the hitting-and-overlapping issue occurs. For long-term solution, the company could consider

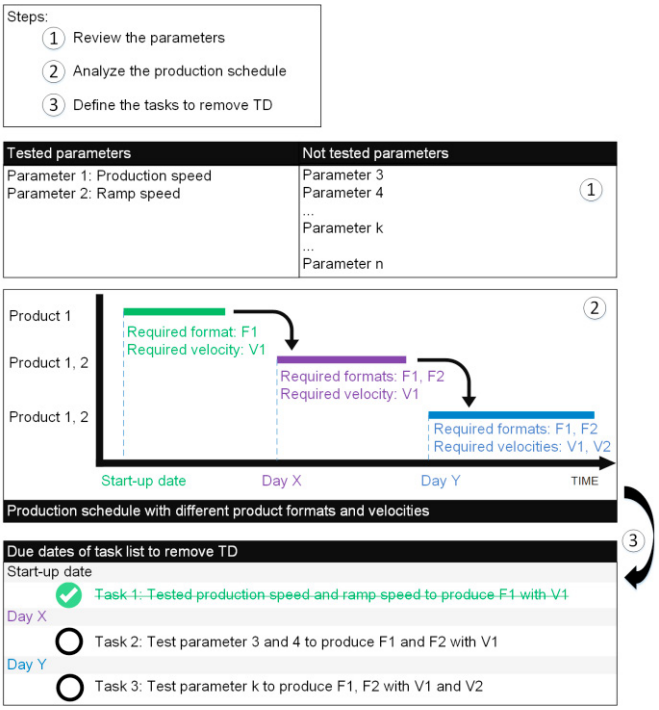


Fig. 7. Short-term recommendation for case B

applying some virtual commissioning techniques before the commissioning phases of other plants. One noteworthy technique is presented in Süß et al. (2016) by researchers from both academia and industry. The virtual commissioning techniques could support the engineers in the preparation for possible situations, which can be extracted from production schedule of new customers. If an issue is found before commissioning phase, the engineers could have enough time to find an optimal solution for it. Hence, TD could be prevented at later phases of aPS. There is an increasing use of virtual commissioning techniques when developing the automated manufacturing plants, according to Süß et al. (2016).

As aforementioned, currently, plant manufacturers are only able to test the systems at customer site. If an issue occurs at another commissioning phase of another customer and it would take time to implement optimal solution or an optimal solution is not available yet, then on-site engineers tend to choose a sub-optimal solution to meet deadline of this short phase. Thus, TD is triggered and an extra effort might be required to remove TD later. When an aPS manufacturer (for instance, company in case B) tries expanding its market boundaries, different requirements from new customers might come to the development team. This means that the development team needs more preparations to the market expansion. Rösch et al. (2014) demonstrated a significant approach to verify faults and reliability of plants. The test cases are generated from timing sequence diagrams. This approach could work at the component level. In case B, the manufacturer met an issue with the timing when the products flow from one to another workstation. The similar issues could occur at new plants. Model-based testing methodology could support the development team to detect these troubles prior to the commissioning phase; because one advantage of this approach is, the timing as well as the diversity of the materials could be defined and verified at early phases of aPS development process. As this technique could support the team cover more cases at early phases, the on-site engineers could meet less problems at commissioning period. This preparation could support aPS manufacturers to prevent the TD. The organizations do not need to spend extra effort for paying interest of TD and could operate more economically. As aforementioned, TD always corresponds to cost. Hence, the model-based testing technique could be considered as another suggestion to the company in case B as well as other plant manufacturers.

5. A DRAFT EVALUATION BASED ON ONLINE SURVEY RESULT

The intention of this step is to construct validity of this study. The research results from case studies could be more objective if they are validated by another source of information – quantitative results from an online survey. The information from an online survey is often gained in a structured form and has a large volume. Only full responses are considered in this evaluation phase. A full response is counted when the participant submitted the answers for all questions. There were 79 full responses. The feedback was received from German

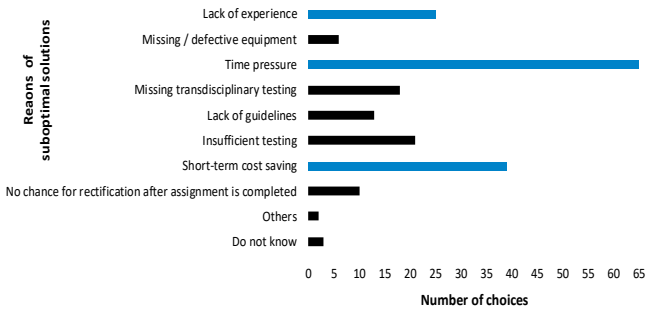


Fig. 8. Most frequent reasons for the choice of a suboptimal solution instead of the most reasonable one (blue columns are top three reasons)

industrial organizations working with aPS at various markets. The participated companies were operating at 40 market domains. Those markets included automotive, medicine, packaging machinery, robotics and automation, etc. The participants were from multiple disciplines such as mechanics, IT, electronics/electrical engineering, etc.).

First, we focus on comparing causes of the suboptimal solutions in the case studies with frequent reasons of suboptimal solutions, identified from the online survey. Regarding the question *"What are the most frequent reasons for the choice of a suboptimal solution instead of the most reasonable one?*", the top three answers were *Time pressure*, *Short-term cost saving* and *Lack of experience*. Each participant could choose more than one answer in this question, since there might be multiple triggers of choosing suboptimal solutions at one company. Fig. 8 illustrates an overview on how the answers were voted by participants. The result of this question can confirm validity of the causes being found in our two case studies. In case A, *Short-term cost saving* and *Lack of experience* reasons triggered the long-term and cross- disciplinary additional effort. In case B, *Time pressure* at commissioning phase led to unscheduled extra effort at next life cycle phases of aPS.

Second, we focus on disciplines which are being affected by additional long term effort (unscheduled cost). Regarding the question *"In which discipline does the long term additional*

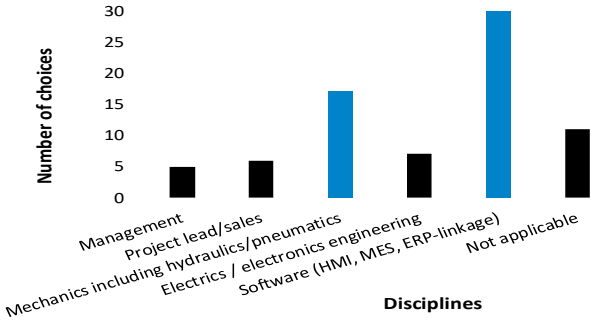


Fig. 9. Discipline in which additional long term effort occurs (software and mechanics disciplines have the highest unscheduled efforts – blue columns)

effort occur?", the informants ranked software discipline has the most occurrences of long term additional effort and mechanics discipline is at the second position (see Fig. 9). Each participant could only choose one answer in this question since we wanted to determine the discipline where has the most additional effort, which comes closest to experience of that participant. The results at software and mechanics disciplines match with the recommendation to case A. It shows that other machine and plant manufacturers participated in the survey might have similar problem to the company in case A. Thus, it is suggested that those manufacturers conduct a review on their open projects, focusing on the tasks of software and mechanics disciplines, to check if TD occurs.

In summary, the quantitative results of the survey would seem to confirm the findings of the case studies, but further analysis is required in the future to validate achieved results.

6. CONCLUSION AND FUTURE WORK

Industrial case studies show TD from mechanical discipline has significant interest to software discipline of aPS. In case cross-disciplinary TD occurs, efforts from multiple disciplines might be required to remove that TD item completely. Cross-life-cycle-phase TD items are also presented. New cross-disciplinary and cross-life-cycle-phase TD items found in case studies are categorized and an enlargement of current TD taxonomy for aPS domain is achieved. The recommendations to remove/prevent TD items in case studies could be generally applied to other machine and plant manufacturers. We also present TD4aPS, an adaption of SWMAT4aPS benchmark process, as a systematic research method for TD in aPS domain. It shows that SWMAT4aPS is extendable and could be adapted to serve further research needs in aPS domain.

For upcoming work, additional evaluation of results from the online survey would be conducted. We plan to collect more examples of TD related to electrical/electronic discipline (e.g. TD from cabinet construction phase or TD from electrical/electronic engineering and software integration phase). We also plan to conduct further research at international aPS companies.

ACKNOWLEDGEMENTS

The authors are grateful to all participated companies for the contributions. We would like to thank anonymous reviewers for their constructive comments and suggestions. We would like to thank Eva-Maria Neumann and Juliane Fischer for the support in the research. We are grateful to Vietnam International Education Development and Vietnamese-German University for the grant of scholarship to Quang Huan Dong under Project 911 - "Training lecturers of Doctor's Degree for universities and colleges for the 2010-2020 period" (Decision No. 911/QD-TTg dated 17/6/2010).

REFERENCES

Avgeriou, P., Kruchten, P., Nord, R. L., Ozkaya, I., Seaman, C. (2016a). Reducing Friction in Software Development. *IEEE Software*, 33(1), pp. 66–73.

Avgeriou, P., Kruchten, P., Ozkaya, I., Seaman, C. (2016b). Managing Technical Debt in Software Engineering. *Dagstuhl Reports*, 6(4), pp. 110–138.

Besker, T., Martini, A., Tichy, M., Bosch, J. (2017). An Investigation of Technical Debt in Automatic Production Systems. In: *Proceedings of the XP2017 Scientific Workshops on - XP '17*. (In press)

Capitán, L., Vogel-Heuser, B. (2017). Metrics for Software Quality in Automated Production Systems as an Indicator for Technical Debt. In: *2017 IEEE International Conference on Automation Science and Engineering (CASE)*. (In press)

Cunningham, W. (1993). The WyCash Portfolio Management System. *ACM SIGPLAN OOPS Messenger*, 4(2), pp. 29–30.

Li, Z., Avgeriou, P., Liang, P. (2015). A Systematic Mapping Study on Technical Debt and Its Management. *Journal of Systems and Software*, 101, pp. 193–220.

Martini, A., Bosch, J. (2016). Architectural Technical Debt in Embedded Systems. *INCOSE International Symposium*, 26(1), pp. 1029–1043.

Rösch, S., Tikhonov, D., Schütz, D., Vogel-Heuser, B. (2014). Model-Based Testing of PLC Software: Test of Plants' Reliability by Using Fault Injection on Component Level. *IFAC Proceedings Volumes*, 47(3), pp. 3509–3515.

Seaman, C., Nord, R., Kruchten, P., Ozkaya, I. (2015). Technical Debt: Beyond Definition to Understanding. *ACM SIGSOFT Software Engineering Notes*, 40(2), pp. 32-34.

Süß, S., Magnus, S., Thron, M., Zipper, H., Odefey, U., Fäßler, V., Strahilov, A., Kłodowski, A., Bär, T., Diedrich, C. (2016). Test Methodology for Virtual Commissioning Based on Behaviour Simulation of Production Systems, in: *International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–9.

Vogel-Heuser, B., Rösch, S. (2015a). Applicability of Technical Debt as a Concept to Understand Obstacles for Evolution of Automated Production Systems. In: *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 127–132.

Vogel-Heuser, B., Rösch, S., Martini, A., Tichy, M. (2015b). Technical Debt in Automated Production Systems. In: *31st International Conference on Software Maintenance and Evolution (ICSME)*, pp. 49–52.

Vogel-Heuser, B., Fischer, J., Feldmann, S., Ulewicz, S. Rösch, S. (2017a). Modularity and architecture of PLC-based software for automated production Systems: An analysis in industrial companies. *Journal of Systems and Software*, 131, pp. 35-62.

Vogel-Heuser, B., Neumann, E.-M. (2017b). Adapting the Concept of Technical Debt to Software of Automated Production Systems Focusing on Fault Handling, Modes of Operation, and Safety Aspects. In: *Proceedings of the 20th World Congress of the International Federation of Automatic Control (IFAC)*. (In press)

Zazworka, N., Vetro', A., Izurieta, C., Wong, S., Cai, Y., Seaman, C., Shull, F. (2014). Comparing Four Approaches for Technical Debt Identification. *Software Quality Journal*, 22(3), pp. 403–426.

Paper2

Production Engineering 13/3 (2019), pp. 273-282.
https://doi.org/10.1007/s11740-019-00897-0

Production Engineering (2019) 13:273–282
https://doi.org/10.1007/s11740-019-00897-0

COMPUTER AIDED ENGINEERING

Technical Debt as indicator for weaknesses in engineering of automated production systems

Quang Huan Dong[1,2] · **Felix Ocker**[1] · **Birgit Vogel-Heuser**[1]

Received: 29 January 2019 / Accepted: 2 April 2019 / Published online: 6 April 2019

Abstract
The concept of Technical Debt describes a situation in which a technical compromise is made despite better knowledge. The survey presented delivers insights on Technical Debt in 48 German companies supplying automated production systems. The participating companies do have some immediate benefits from taking Technical Debt under time pressure, but encounter a significant higher long-term additional effort to recover from technical debt. However, awareness for Technical Debt at these companies is low. Therefore, the automated production system manufacturers need to keep a closer eye on expenditure for Technical Debt. The developed survey can be used as a self-assessment method for other companies to compare their results with the average results from this survey.

Keywords Technical debt · Cross-disciplinary engineering · Automated production systems · Management

1 Introduction and motivation

Technical Debt (TD) describes a situation in which a technical compromise is made, e.g., delivering not-quite-right code in order to meet an urgent deadline [1]. The technical compromise chosen can yield a short-term benefit (TD benefit) but may cause a long-term negative impact (TD interest) on the system quality or the productivity of engineers [2]. The concept of TD can be transferred to the development of automated Production Systems (aPS), which are used nowadays to create products in various sectors, including automated packaging, pharmaceutical production or food processing. aPS are specific classes of mechatronic systems. Typically, aPS are designed by engineers from the three disciplines mechanical, electrical and software. Other disciplines such as hydraulic, pneumatic, sensor or drive technology may also get involved whenever needed. There are strong interdependencies between all those disciplines. Analysing TD in the software discipline at one aPS company, Besker et al. [3] suggested to include mechanical and electrical disciplines additionally to software.

A survey is conducted to address uncovered aspects in prior work [4] such as cost, as well as consciousness of TD in the aPS domain. The study considers the views from management and specialists as well as different types of aPS manufacturers including inputs from software, electrical, and mechanical hardware personnel.

The main contribution of this paper is the first quantitative survey to determine the state-of-the-practice regarding TD in the aPS domain. The paper thus provides a sound basis for TD management in order to reduce costs in engineering and manufacturing.

2 Technical Debt in automated production systems

A classification of TD types according to different causes for general software systems was presented by Li et al. [2]. Some significant causes are improper architecture, test, documentation, and, most studied, code TD. The work of Avgeriou et al. [5] indicates that TD always relates to cost. Failure to monitor TD has resulted in unexpectedly large

✉ Quang Huan Dong
huan.dong@tum.de

✉ Felix Ocker
felix.ocker@tum.de

✉ Birgit Vogel-Heuser
vogel-heuser@tum.de

[1] Institute of Automation and Information Systems, Technical University of Munich, Munich, Germany

[2] Vietnamese-German University, Thu Dau Mot City, Vietnam

cost overruns in many software development projects [6]. TD might have influences on the planned, reported, and actual product delivery date of a project [7–9], profitability of the organization [10–12]. However, sometimes the software engineers request investments to remove TD, but after inquiring about their business value, the executives might decline [13]. In addition, the individuals choosing to incur TD (e.g., designers) could be different from those responsible for recovering from the debt (e.g., maintenance staff) [14] [15].

In the domain of software engineering for embedded systems, which is more similar to the aPS domain, Martini et al. [16] investigated reasons for TD. They identified, e.g., *Time pressure*, *Lack of knowledge* and *Parallel development*. However, *Parallel development* as reason for TD did not yet cover the factor of cross-disciplinarity. For a specific case, Martini et al. [17] reported that improving modularity can reduce architectural TD in the software.

In the aPS domain, Besker et al. [3] studied the work of software developers at one aPS company in Scandinavia and identified that they spend "… *quite a lot of resources in paying the interest on Technical Debt, on average 32% of the development time*". The estimated amount of TD was confirmed by managers. Vogel-Heuser et al. [18] identified that, on average, plant manufacturers have larger software projects compared to the projects of machine manufacturers. Taking this difference into account, the ways of coping with TD should be analysed, depending on the company type – plant manufacturer or machine manufacturer (*RQ1*). Furthermore, it is unclear how project characteristics affect TD and the awareness for TD (*RQ2*). Motived by the interdisciplinarity of aPS development [14], it is necessary to study which discipline and phase of the life cycle are affected by decisions of taking TD (*RQ3*). In prior work [4], some important aspects such as cost saved, long-term additional cost and consciousness of TD were not covered. In addition, it is important to analyse how the management and specialists rank the amount of TD with regard to the disciplines involved and the causes of TD (*RQ4*).

The survey by Schuh et al. [19] revealed some major challenges for manufacturing companies. In another survey, also with German companies from the manufacturing industry, Schuh et al. [20] identified some factors contributing to the performance of the systems developed. However, TD aspects have not yet been addressed, since those two surveys focused either on upcoming technologies [19] or on information exchange between service and development [20]. According to Vogel-Heuser et al. [21], maintainability is a prerequisite for the evolution of aPS, which have an especially long lifetime. In order to improve maintainability, high modularity of aPS software is a critical factor [18]. Being good at modularity would enable reusability and consequently higher efficiency in development as well as higher software quality. However, insufficient management of TD may destroy modularity. Unfortunately, in available surveys, TD was either not considered [19] [20] [21] or TD was only analysed regarding modularity of a specific software at one company close to the aPS domain [17]. As far as we are aware, no other studies have been undertaken to explore reusability, modularity, and TD in different disciplines of the aPS domain. Thus, the effect of reusability and modularity on TD in different disciplines of aPS companies should be analysed (*RQ5*).

In summary, the aPS domain lacks a quantitative study regarding TD. Moreover, the study shall consider the compounds of TD occurrence such as (1) project types/scopes; (2) types of aPS manufacturers; and (3) the perspectives of management and specialists. This research aims to fill this gap in order to gain broader knowledge about TD and the state of the practice in the aPS domain. Based on that, future work can develop proper TD management activities while considering multiple disciplines. Thus, this work forms the basis for a potential increase in cost effectiveness in the aPS domain.

3 Research questions and hypotheses

Following the idea of Li et al. [2], we identify typical settings, in which TD occurs and which constraints, situations, and people or disciplines are involved. The in-depth analysis explores whether specific types of companies or projects are more in danger to encounter TD than others. We also investigate the effect of TD on various disciplines. Saved cost and disciplines having the largest benefit from taking TD are found. Subsequent to the findings from Vogel-Heuser et al. [18], the study includes TD on different levels of modularity and reusability. It is assumed that a high level of modularity or reusability leads to low TD.

Twelve hypotheses were formulated for five research questions in different contexts (e.g., company/project to discipline).

At company/project level, TD can be expected to affect larger projects more, which prevail in plant manufacturing. This is because larger projects are more complex, less transparent, and harder to manage. Nowadays, to serve diverse customer wishes, companies may need to work on a wide range of project scopes. However, typically, plant manufacturer projects are larger in scope than those of machine manufacturers (*H1.1*). This is expected to lead to different TD awareness levels for plant manufacturers and machine manufacturers (*H1.2*). As Besker et al. [3] pointed out, significant additional cost is required due to insufficient attention and management of TD. However, independent of the project's kind, it is unclear what short-term costs are saved, which long-term additional cost are

caused, and what the awareness level is. The general projects, which are not adapted to customer specific requirements, require solutions that are more general, in order to satisfy all customers' requirements. Hence, different kinds of projects might differ in TD. On the one hand, general projects might have less TD benefit (*H2.1*) and more TD awareness (*H2.2*). On the other hand, customer specific projects suffer more from TD interest (*H2.3*). In contrast to Besker et al. [3], who conducted their survey at one company, this survey studies 48 companies.

At the discipline level, besides the complex dependencies, the start time of each discipline on the engineering timeline is different [14]. Usually, the development starts with the mechanical engineering discipline, which creates the construction plan of the mechanical parts. Based on this construction plan and component lists of sensors, actuators and valves, the electrical engineers design the electrical system. This includes the corresponding circuit diagram, the connection of sensors and actuators to the programmable logic controller and the task of distributing power. Documents from both disciplines are then forwarded to the software engineers, who use them to develop the control software for the soft or hardware programmable logic controllers [14]. Due to the sequence, the departments responsible for TD might choose sub-optimal solutions, which TD interest is induced to other departments. Thus, the departments responsible for TD are affected less by their own decisions (*H3.1*). In case one department has lots of influence (e.g. is larger than the others) in a project, it might be able to force TD on other departments (*H3.2*). Up to now, it is unclear how management and specialists perceive the amount of TD benefit and interest at disciplines (*H4.1*). In different interviews, software engineering is the often blamed department for not finishing their task and that we want to study with this survey whether software is the most benefited as well as the most affected discipline (*H4.2*). Furthermore, from the views of management and specialists, *Time pressure* is the most common reason for TD (*H4.3*). Besides the reasons found in Martini et al. [16], the research would include the reasons relating to cross-disciplinary development such as equipment unavailability or missing transdisciplinary.

At different disciplines, there might be different experience in development methods and achieve different modularity and reusability level (*H5.1*) in order to cope with TD. Hence, different modularity and reusability levels might lead to different levels of TD (*H5.2*).

4 Method and threats to validity

In this section, the method is described and potential threats to validity are reported.

5 Method

A survey was conducted with German machine and plant manufacturers from June until October 2017. The survey was distributed by an independent publishing service (Vogel Business Media GmbH & Co. KG). Eighty complete responses were collected from 48 companies. The participating companies operate in various markets such as water treatment, medicine, automotive, printing, et cetera. The percentages of participants involved in electrical, software, and mechanical disciplines are 69, 49 and 17% respectively. Details of the questions used in this study are available online [22]. The originally German questionnaire was translated for this paper. Hereafter, Q#[number] denotes a question, which has the according ordinate number in the translation. Based on the discussions between authors, the answers of each question were ranked from zero to five (maximum score), where applicable. For example, the scores at Q#2.6 (short-term cost saved) are zero (for "0%" answer), 1.25 (for "1–15%" answer), 2.5 (for "16-30%" answer), 3.75 (for "31–60%" answer), 5 (for "> 60%" answer). Another example is the scoring for Q#2.9 (TD awareness), which ranges from zero (for "No, not at all"), to 1.25 (for "In selected departments"), to 2.5 (for "At the project lead"), to 3.75 (for "Within the management") to 5 (for "In the whole company"). To ensure validity of the results, "could not determine" or "not applicable" (n.a.) answers were not scored and excluded from the analysis.

In this paper, besides the three disciplines mentioned (i.e. mechanical, electrical and software parts), the studied systems also include hydraulic, pneumatic, sensor or drive technology. Hereafter, the term mechanics includes mechanical, hydraulic and pneumatic disciplines and the term electrical includes electrical, sensor and drive technology.

Besides TD, the survey contains questions aimed at other aspects such as tools for version/variant management or maturity in software exchange. As the scope in this paper is limited to TD, only questions related to the TD perspective are considered. In total, thirteen questions provide valuable information in the context of TD: one question about the profile of participants (Q#1.4), four questions about characteristics of projects (Q#1.6, Q#1.7, Q#1.8, Q#1.14), one question about reusability (Q#1.15), one question about modularity (Q#2.5) and six questions about TD itself (Q#2.6, Q#2.6.1, Q#2.7, Q#2.8, Q#2.8.1 and Q#2.9). To answer a research question, it might be necessary to select and combine the results of individual questions. For example, Q#1.7, Q#1.14 and Q#2.9 are used to answer RQ1. A correlation coefficient analysis was performed with MATLAB in order to assess the strength of relations between the scores in TD, modularity or reusability questions.

5.1 Threats to validity

First, there might be a bias when deriving the research questions and hypotheses. To reduce this bias, the research questions follow the ideas and suggestions from recent publications as well as feedback from discussions and workshops with experts from industry.

Secondly, there might be a bias when developing questions and their answers in the survey, especially questions related to the TD perspective, since the concept of TD is not yet well known in the aPS domain. To mitigate this threat, two interviews were conducted to test the survey with experts who already had some knowledge about TD and who are working with aPS. These interviewees were a developer lead and a quality assurance engineer from two different companies. To reduce the time to fill in the survey, the terms used in six questions about TD were carefully revised so that the participants could provide the answers quickly. For example, the term "long term additional effort" was selected instead of "effort to pay TD interest". The term TD only appears at the last question (Q#2.9) for TD aspect. In Q#2.9, textual options are preferred to score/percentage options since the textual ones can describe the TD consciousness level better.

Third, the participants and their companies were not selected by the authors, but by an independent business media partner. Thus, we do not know the company names. There is a threat that if participants did not provide exact answers, we will not be able to figure this out. Nevertheless, diversification of the responses can be reached as the survey was sent to a broad range of plant and machine manufacturers, which are in the network of the partner. In addition, the method using such a survey has been proven to show reasonable results as it has been compared in prior studies to interview results.

Fourth, during the analysis phase, the survey results were analysed independently by various employees of our institute, were consolidated, and discussed with domain experts in order to reduce errors in the results' interpretation.

In Sect. 5.6, the validity of each finding is made clear by use of a traffic light indicator.

6 Results

This section presents results of the research questions individually and closes with a summary of our findings and an assessment of the findings' validity.

6.1 How do plant manufacturers and machine manufacturers cope with Technical Debt? (Research Question 1)

The amount of plant engineering, special purpose machinery or serial machinery projects is asked in Q#1.7 as a pre-processing step. If a participant could not determine the amount, the response is excluded from analysis. Machine manufacturers create individual machines. Plant manufacturers build plants, which include individual machines. Companies were categorized as machine manufacturers or plant manufacturers according to the majority of their projects. Question Q#1.14 addresses the scopes of the primarily created projects (see Fig. 1). As can be seen in Fig. 1, only a miniscule number of plant manufactures provides *Single modules (2)* and a small number produces *Single modules (2)* and *Parts of a machine (3)*. This means, that several plant manufactures also offer parts of machines. Unsurprisingly, the typical scope of plant manufacturers' projects (4.22) is bigger than the mean scope of machine manufacturers' projects (3.29) and the upper quartile of machine manufactures creates *Complete machine (4)*. We conclude that *H1.1 is true*.

The different scopes of projects might lead to different TD attention because managing large scope projects is more complex. It can be observed that plant manufacturers score better TD awareness level than machine manufacturers (mean 2.50 > mean 1.12) (Fig. 2). *H1.2 is true*. Nevertheless, TD at both plant manufacturers and machine manufacturers might have been a good choice because values for short-term cost savings are better than the values of long-term additional cost (plant manufacturers: 2.14 > 1.94; machine manufacturers: 2.60 > 1.82).

6.2 How do project characteristics affect Technical Debt and the awareness for Technical Debt? (Research Question 2)

The amount of specific work (e.g. design of a new machine or adaptation of an existing one to a specific customer wish) might vary in different project kinds. aPS from *Projects*

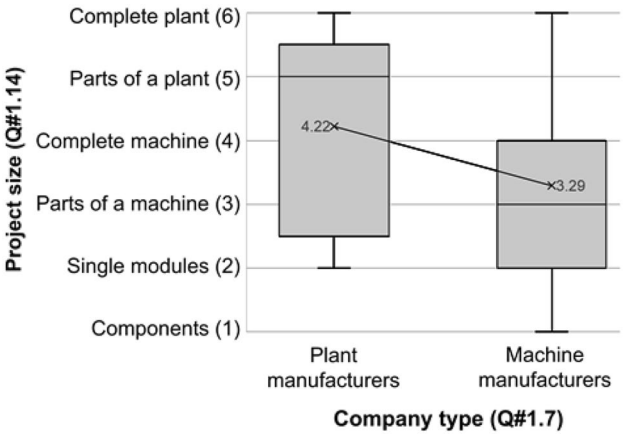


Fig. 1 On average plant manufacturers show larger project scope than machine manufacturers (Q#1.14, 33 responses)

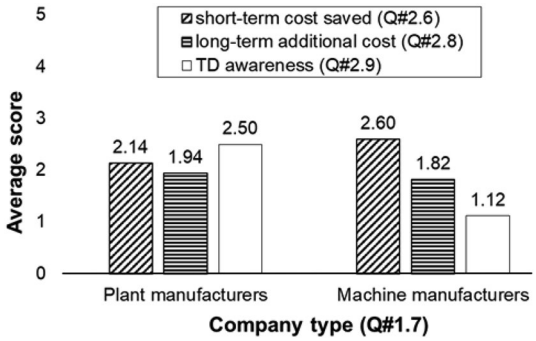


Fig. 2 Plant manufacturers have better TD awareness than machine manufacturers (Q#2.9, 33 responses)

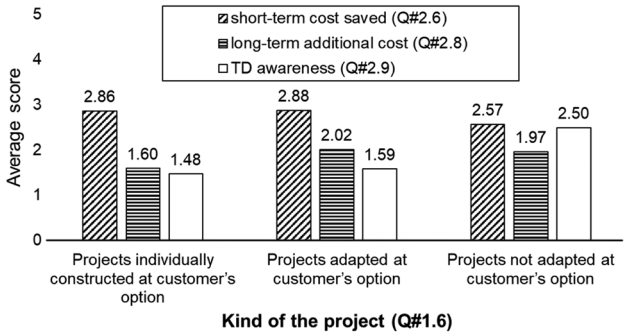


Fig. 3 Project characteristics and TD (57 responses)

individually constructed at customer's option (i.e. customer specific projects) are designed individually and developed for specific customer requests. aPS from *Projects not adapted at customer's option* (i.e. not-adapted projects) are designed and developed for multiple customers. The scope of *Projects adapted at customer's option* (i.e. partially adapted projects) stays in between the two above project kinds. The shares of the different kinds of projects in the total amount of projects are obtained at question Q#1.6. The shares are used to identify the project kind, which has majority at the company of each participant. Figure 3 presents benefit, interest as well as attention of TD at different project kinds.

Regarding TD benefit, customer specific projects (2.86) and partially adapted projects (2.88) have better scores than general (not-adapted) projects (2.57). *H2.1 is partially true* as the score gap is small. One might expect that the customer specific ones score much more on TD benefit, because they allow less standardization. However, the result is not exactly as hypothesized.

Customer specific projects and partially adapted projects have quite low TD awareness levels (1.48 and 1.59 point in respectively). Not-adapted projects have higher TD awareness (2.5 point) compared to the two kinds of projects above. *H2.2 is true*. It could be explained that customer specific projects would require solutions, which are more general for all customers and, thus, might affect TD recovery strategies and levels of consciousness for TD.

For TD interest, customer specific projects (1.60) have the lowest score compared to general projects (1.97) and partially adapted projects (2.02). Hence, *H2.3 is true*. It seems that there is a significant amount of TD at customer specific projects.

6.3 Which discipline and which phase of the life cycle are affected by decisions of taking Technical Debt? (Research Question 3)

RQ3 analyses whether the departments responsible for TD are not affected by their own decisions (inverse relation between Q#2.6.1 and Q#2.8.1). Question Q#2.6.1 asked the participants, which discipline often takes TD benefit (i.e. effort is saved) and Q#2.8.1 asks for the discipline which has to pay the respective TD interest (i.e. long-term additional effort). Question Q#1.8 enquires the number of engineers from each discipline that are involved in an average project. The number of engineers is used to check whether a discipline has a majority in the project. The distribution of TD interest when one department takes TD benefit is illustrated in Fig. 4.

When the discipline mechanics takes TD benefit, 89% TD interest occurs at mechanics itself (Fig. 4). The majority of employees in 50% of these projects are mechanical

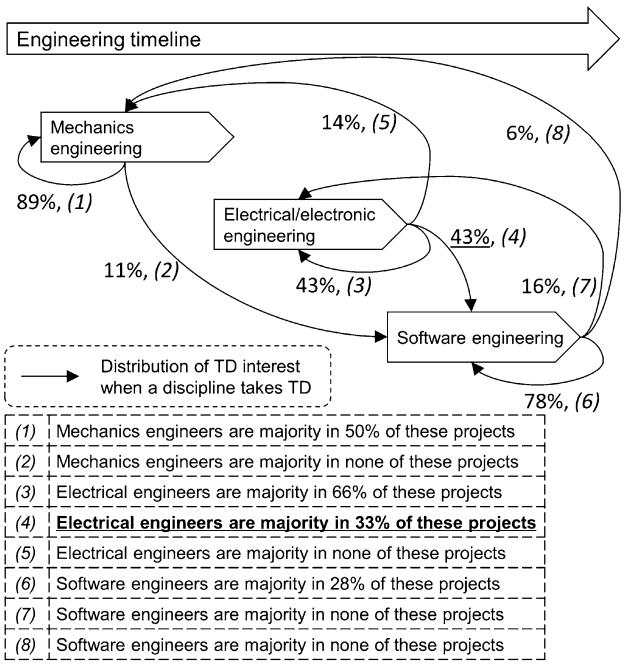


Fig. 4 When taking TD, electrical discipline induces significant TD interest in software discipline (Q#2.6.1, Q#2.8.1, 35 responses)

engineers. The remaining TD interest (11%) has to be paid by the software discipline. Mechanical engineers are majority in none of the projects, which put TD interest on the software discipline. Overall, mechanics is affected only by its own decisions and mechanical engineers are not felt to force TD on others. A similar situation occurs in the software discipline.

An interesting result is collected regarding the electrical discipline. When the electrical discipline takes TD benefit, 43% of TD interest is shifted to the software discipline and electrical engineers form the majority in 33% of these projects (cf. (4), Fig. 4). 43% of TD interest occurs in electrics and electrical engineers are the majority in 66% of these projects (cf. (3), Fig. 4). 14% of TD interest occurs at mechanics and electrical engineers are the majority in none of these projects (cf. (5), Fig. 4). Overall, the electrical discipline is only partially affected by its own decisions. When electrical engineers form the majority in a project, sometimes (33% of projects which electrical engineers are majority) they force TD interest on the software discipline. The electrical discipline causes significant TD interest on software discipline. In conclusion, *H3.1 is partially true*.

In summary, mechanics, and software take both TD benefit and TD interest. The electrical discipline takes TD benefit and causes significant TD interest on the software discipline. When mechanics or software engineers form the majority in a project, they do not force TD on others. When electrical engineers are the majority in a project, sometimes they do force TD on software engineers. In addition, as aPS software development highly depends on the electrical development (and mechanics development as well), it could be the main reason that the software discipline often takes TD interest when the electrical discipline takes TD benefit.

6.4 How do management and specialists rank the amount of Technical Debt at disciplines and the causes of Technical Debt? (Research Question 4)

Question Q#1.4 is used to classify the respondents. Nearly half of respondents (45%) are in leadership or management positions (e.g., director, head or group leader). The percentage of respondents as specialist is 40%. The remaining respondents (15%) did not reveal their positions. Thus, the results can roughly be divided into two views: management and specialists. In this part of the analysis, only the responses from the respondents who revealed their positions are counted.

Question Q#2.6 studies the average effort that can be saved in the short term by implementing a sub-optimal solution in comparison to implementing the most reasonable one. The most votes are for 16-30%, which has 39% votes from management and 38% votes from specialist (Fig. 5).

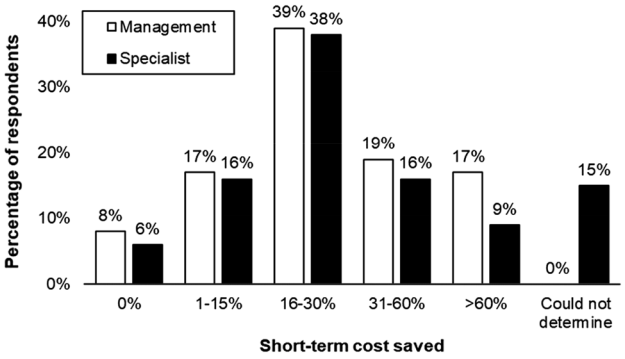


Fig. 5 Both management and specialists have the most votes for 16-30% of effort can be saved if TD is taken (Q#2.6, 68 responses)

A wise Gaussian distribution can be observed that management and specialist have similar rates for the benefit of TD, except at > 60% which has more votes from management (17%) than from specialist (9%).

Question Q#2.8 studies the long-term additional effort caused by the implementation of a sub-optimal solution compared to the implementation of the most reasonable solution. Both management and specialist have high number of votes at large TD interest (31-60% and > 60%) (Fig. 6). *H4.1 is true*. It should be noted that specialists have higher ratings (focusing on 16-30%, 31-60% and > 60%) than management.

The result of question Q#2.6.1 indicates disciplines with the biggest potentials for savings. Both management (31%) and specialists (34%) agree on the software discipline (cp. (1) in Fig. 7). There is still a significant gap between the votes of management and specialists at each discipline. For example, project lead/sales has 19% of management and 25% votes from specialists, which result in a delta of 6. The average delta is 9 and yet there is a significant number of specialists (23%) who could not determine the discipline.

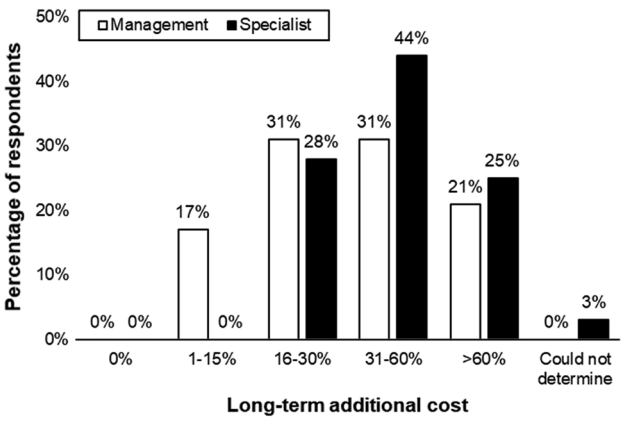


Fig. 6 TD requires significant long-term additional effort (Q#2.8, 68 responses)

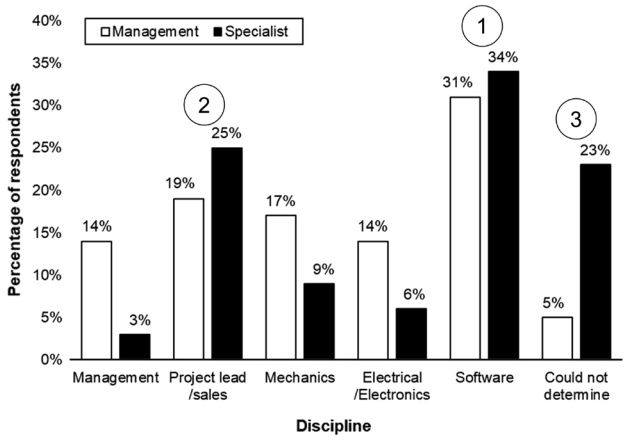


Fig. 7 Software discipline with the most benefit from taking TD (Q#2.6.1, 68 responses)

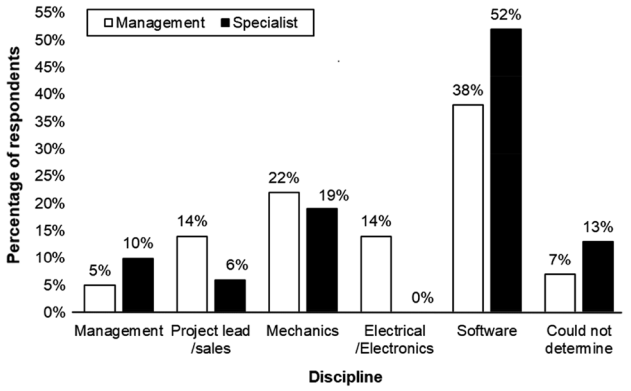


Fig. 8 Software discipline with the most impacts from TD (Q#2.8.1, 68 responses)

Regarding the discipline that has to pay the most TD interest (Q#2.8.1), the software discipline gets the most votes from both management (38%) and specialists (52%) (Fig. 8). Comparing the results of mechanics, electrics and software, the electrical discipline has the least votes (14% from management and 0% from specialist). Overall, the software discipline has the most votes for TD benefit (Fig. 6) as well as the most votes for TD interest (Fig. 8), from both management and specialists. Overall, *H4.2 is true*.

Question Q#2.7 studies the reasons for the choice of a sub-optimal solution instead of the most reasonable one. One participant can vote for multiple reasons as several causes of TD might apply at the same time. Overall, both management and specialists vote similarly for all reasons (Fig. 9). The highest votes at *Time pressure* from both management (83%) as well as specialists (88%) confirm the finding of the work from Martini et al. [16] in the embedded software engineering domain, where pressure to on time delivery is a significant TD trigger, *H4.3 is true*.

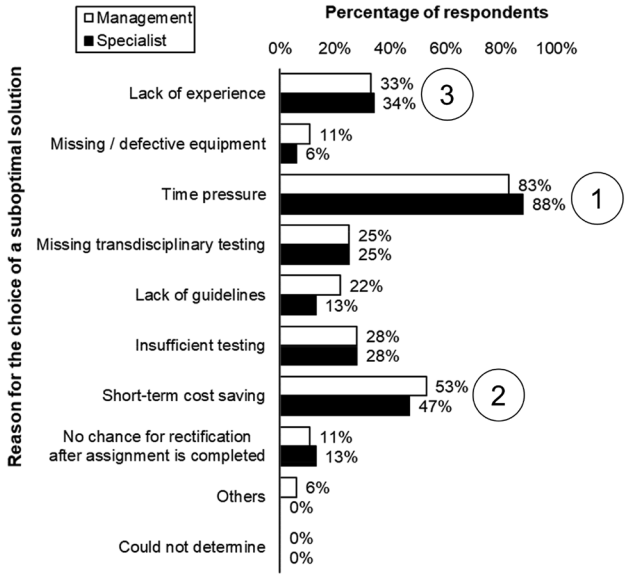


Fig. 9 Causes of TD (Q#2.7, 68 responses)

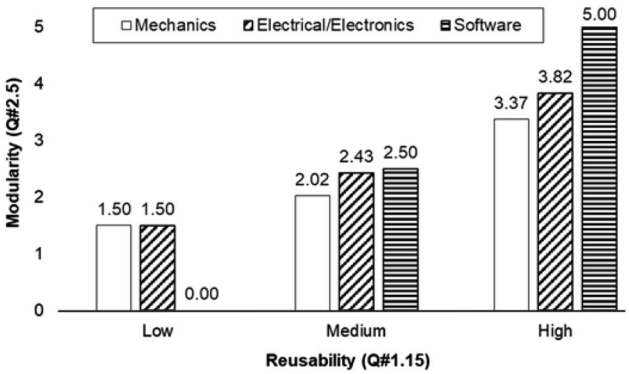


Fig. 10 High modularity enables high reusability (67 responses)

6.5 How do different reusability and modularity levels affect Technical Debt? (Research Question 5)

Different reusability levels are obtained from question Q#1.15. Scores of modularity (Q#2.5) vary from 2 (*Project specific share*), 3 (*Libraries/templates*), 5 (*Both*) and 0 (*Neither nor*). *Could not determine* responses are excluded from the analysis. Figure 10 presents the modularity at different reusability levels. It can be observed that at each discipline, the different ratings on modularity lead to different reusability levels. Hence, *H5.1 is true*. It can be explained that, most of the time, system needs to be well modularized in order to be good at reusability. In addition, a medium correlation ($r = 0.693$) is identified between modularity of the software discipline and modularity in electrics/electronics.

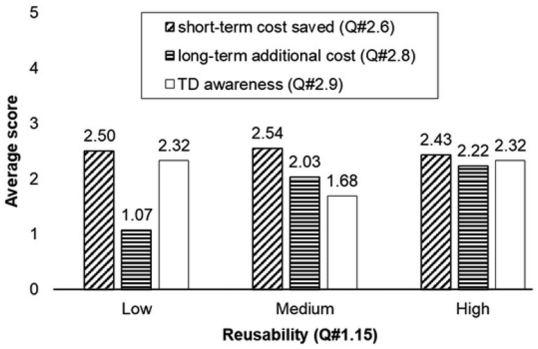


Fig. 11 High reusability leads to low TD interest (56 responses)

TD benefit, interest, and consciousness at different reusability levels are illustrated in Fig. 11. An interesting result is that companies, which score low reusability also score low TD interest (1.07). Thus, different levels of reusability might lead to different levels of TD. *H5.1 is partially true* since TD benefit and attention are similar at different reusability levels. One explanation could be that TD interest is low with low proficiency to reuse, not because the system is not reusable, but because the company does not reuse it and therefore they do not pay long-term interest on reused system with TD.

6.6 Summary of Findings and Their Validity

The research questions, related hypotheses, and validity are summarized in Fig. 12. The result from *H1.1* confirms the result in Vogel-Heuser et al. [18], with a larger group of companies. Although larger project scopes lead to higher awareness for TD (*H1.2*), it might be argued that larger projects might have developed better TD recovery strategies; however, the TD recovery strategies remain unclear at these companies and should be checked in future research.

Regarding *H3.2,* besides majority of engineers, the influences "power" might be enrooted in other factors such as prestige or management support. It would be interesting to check the influences from those factors in future research, too. As aPS software development often starts after mechanics and electrical development (due to dependencies), "quick-and-dirty" solutions might be implemented in software, in case the project deadline is close.

As a medium correlation between modularity of electrical and software disciplines is identified, it seems that mechanical engineering just does not have as many dependencies with electrics as electrics have with software engineering. Furthermore, bad mechanical engineering decisions might not be perceived as TD, but rather as a boundary condition.

Regarding *H4.1* to *H4.3*, management and specialists rank the amount of TD quite similar. This confirms the findings of Besker et al. [3]. For *H4.2*, although the software discipline has the highest votes from both management (30.56%) and specialists (34.38%) for disciplines taking TD benefit, it is assumed that this perception might be from bad bug fixes from the software engineers. It could be also the case because most people crossing this box are not from the software department. Therefore, the reasons for this perception should be taken into account in future research.

Regarding *H5.2*, with similar TD consciousness, the factor modularity might play an important role. Mature modularity solutions lead to higher reusability (*H5.1*), which in turn reduces TD interest (mean 2.03 and 2.22) in comparison to those cases with low reusability (mean 1.07).

7 Conclusion and outlook

This study uncovered that the decision of taking TD benefit by the electrical discipline causes significant TD interest (43%) in the software discipline. The results reveal important details to leverage the transparency of unscheduled cost between different disciplines in a TD perspective. TD has a significant impact on the overall cost for aPS. However, TD awareness at these companies is low. Therefore, both plant manufacturers and machine manufacturers should monitor costs for TD more closely. The developed survey can be used as a self-assessment method for other companies. Thereby, the average results from this study can serve as a benchmark.

Future work should study TD recovery strategies, which aPS manufacturers use to cope with cross-disciplinary TD since no existing TD management approach has deeply discussed or investigated a utilization in actual cases [2], and furthermore, in a cross-disciplinary environment such as aPS development.

Research question	Hypothesis	Findings		Validity
RQ1 How do plant manufacturers and machine manufacturers companies cope with TD?	H1.1: On average, plant manufacturers and machine manufacturers show different project scopes.	■	Scope of primary software project (machine manufacturers 3.29 < plant manufacturers 4.22) (cp. Fig. 1, Q#1.14)	●
	H1.2: Large project scopes have better level of awareness for TD.	■	Plant manufacturers, which has larger project scopes, has better TD awareness than machine manufacturers (2.50 and 1.12) (cp. Fig. 2, Q#2.9)	●
RQ2 How do project characteristics affect TD and the awareness for TD?	H2.1: General projects have less TD benefit than other kinds of projects.	◧	TD benefit at projects (General 2.57 < Customer specific 2.86 < Partially adapted 2.88) (cp. Fig. 3, Q#2.6)	◑
	H2.2: General projects have better TD awareness.	■	TD awareness at projects (Not-adapted projects 2.50 > Partially adapted projects 1.59 > Customer specific projects) (cp. Fig. 3, Q#2.9)	◑
	H2.3: Customer specific projects are more suffered from TD interest.	■	TD interest at projects (Customer specific 1.60 < General 1.97 < Partially adapted 2.02) (cp. Fig. 3, Q#2.8)	◑
RQ3 Which discipline and which phase of the life cycle are affected by decisions of taking TD?	H3.1: The departments responsible for TD are affected less by their own decisions.	◧	When electrical discipline takes TD benefit, 43% TD interest comes to software discipline (cp. Fig. 4, Q#2.6.1 and Q#2.8.1)	●
	H3.2: Also, in a project, if one department has lots of influences (e.g. larger than the others), it might be able to force TD on other departments.	◧	In 33% projects which electrical engineers are majority, electrical engineers do force TD interest on software discipline (cp. Fig. 4 – (4), Q#2.6.1 and Q#2.8.1)	●
RQ4 How do management and specialists rank the amount of TD at disciplines and the causes of TD?	H4.1: Management and specialists perceive the amount of TD benefit and interest similarly.	■	Both management (39%) and specialist (38%) have the highest votes for 13-30% as the amount of TD benefit (cp. Fig. 5, Q#2.6)	●
			Both management (31%) and specialist (44%) have the highest votes for 31-60% as the amount of TD interest (cp. Fig. 6, Q#2.8)	●
	H4.2: Software is the most benefited as well as the most affected discipline.	■	The most benefited from taking TD is software discipline, which has most votes from both management (31%) and specialists (34%) (cp. Fig. 7, Q#2.6.1)	●
			Both management (38%) and specialist (52%) have the highest votes for software as the most affected discipline (cp. Fig. 8, Q#2.8.1)	●
	H4.3: Time pressure is the most often TD reason.	■	Both management (83%) and specialist (88%) have the highest votes for Time pressure (cp. Fig. 9, Q#2.7)	●
RQ5 How do different reusability and modularity levels affect TD?	H5.1: Different modularity levels lead to different reusability levels.	■	Regarding mechanics discipline, modularity at low reusability level starts from low (mean 1.50) to medium (mean 2.02) at medium reusability, then high (mean 3.37) at high reusability. Similar trends are also observed at electrical and software (cp. Fig. 10, Q#1.15, Q#2.5)	●
	H5.2: Different reusability levels lead to different amount of TD.	◧	Low reusability has low score TD interest (mean 1.07) than medium reusability (mean 2.03) and high reusability (2.22). However, TD benefit and awareness are similar at different reusability levels (cp. Fig. 11, Q#1.15, Q#2.6.1, Q#2.8 and Q#2.9)	◑
■ = true, ◧ = partially true, □ = false				
● = high (based on >= 80% responses), ◑ = medium (based on >=50% and <80% responses), ○ = low (based on <50%)				

Fig. 12 Research questions and hypothesis including findings

Acknowledgements The authors thank all companies that have participated in the survey for their contributions. Also, we thank Huaxia Li, Iris Weiß, and Juliane Fischer for their support in this research. Finally, we thank the Vietnam International Education Development and Vietnamese-German University for granting Quang Huan Dong a scholarship under Project 911—"Training lecturers of Doctor's Degree for universities and colleges for the 2010–2020 period" (Decision No. 771/QD-BGDDT dated 14/03/2017).

References

1. Cunningham W (1993) The WyCash portfolio management system. ACM SIGPLAN OOPS Messenger 4(2):29–30
2. Li Z, Avgeriou P, Liang P (2015) A systematic mapping study on technical debt and its management. J Syst Softw 101:193–220
3. Besker T, Martini A, Bosch J, Tichy M (2017) An investigation of technical debt in automatic production systems. In: Proceedings of the XP2017 Scientific Workshops on XP'17, pp 1–7
4. Dong QH, Vogel-Heuser B (2018) Cross-disciplinary and cross-life-cycle-phase Technical Debt in automated Production Systems: two industrial case studies and a survey. IFAC-PapersOnLine 51(11):1192–1199
5. Avgeriou P, Kruchten P, Ozkaya I, Seaman C (2016) Managing Technical Debt in software engineering (Dagstuhl Seminar 16162). Dagstuhl Rep 6(4):110–138
6. Guo Y et al. (2011) Tracking technical debt—an exploratory case study. In: 2011 27th IEEE international conference on software maintenance (ICSM), pp 528–531
7. Fairley RE, Willshire MJ (2017) Better now than later: managing technical debt in systems development. Computer 50(5):80–87
8. Ramasubbu N, Kemerer CF (2014) Managing technical debt in enterprise software packages. IEEE Trans Softw Eng 40(8):758–772
9. Holvitie J, Leppanen V (2015) Examining technical debt accumulation in software implementations. Int J Softw Eng Appl 9(6):109–124
10. Kruchten P, Nord RL, Ozkaya I (2012) Technical debt: from metaphor to theory and practice. IEEE Softw 29(6):18–21
11. Ampatzoglou A, Ampatzoglou A, Chatzigeorgiou A, Avgeriou P (2015) The financial aspect of managing technical debt: a systematic literature review. Inf Softw Technol 64:52–73
12. Snipes W, Robinson B, Guo Y, Seaman C (2012) Defining the decision factors for managing defects: a technical debt perspective. In: 2012 Workshop on managing technical debt (MTD), pp 54–60
13. Ozkaya I, Kruchten P, Nord R, Brown N (2012) Second international workshop on managing technical debt. In: Proceeding of the 33rd international conference on software engineering—ICSE'11, p 1212
14. Vogel-Heuser B, Rösch S, Martini A, Tichy M (2015) Technical debt in automated production systems. In: 2015 IEEE 7th workshop on managing technical debt (MTD), pp 49–52
15. Martini A, Bosch J (2015) The danger of architectural technical debt: contagious debt and vicious circles. In: 2015 12th Working IEEE/IFIP conference on software architecture, pp 1–10
16. Martini A, Bosch J, Chaudron M (2015) Investigating architectural Technical Debt accumulation and refactoring over time: a multiple-case study. Inf Softw Technol 67:237–253
17. Martini A, Sikander E, Medlani N (2016) Estimating and quantifying the benefits of refactoring to improve a component modularity: a case study. In: 2016 42th Euromicro conference on software engineering and advanced applications (SEAA), pp 92–99
18. Vogel-Heuser B, Fischer J, Feldmann S, Ulewicz S, Rösch S (2017) Modularity and architecture of PLC-based software for automated production systems: an analysis in industrial companies. J Syst Softw 131:35–62
19. Schuh G et al (2011) Technology roadmapping for the production in high-wage countries. Prod Eng 5(4):463–473
20. Schuh G, Gudergan G, Feige BA, Buschmeyer A, Krechting D (2015) Business transformation in the manufacturing industry—how information acquisition, analysis, usage and distribution affects the success of lifecycle-product-service-systems. Proc CIRP 30:335–340
21. Vogel-Heuser B, Ocker F (2018) Maintainability and evolvability of control software in machine and plant manufacturing—an industrial survey. Control Eng Pract 80:157–173
22. Survey on Technical Debt in construction—a lever for more transparency for unscheduled efforts during cooperation. https://mediatum.ub.tum.de/1472174?show_id=1472171&style=full_text. Accessed: 22 Mar 2019 **(online)**

Paper3

IFAC-PapersOnLine 54/1 (2021), pp. 912-917.
https://doi.org/10.1016/j.ifacol.2021.08.108

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-1 (2021) 912–917

Modelling technical compromises in electronics manufacturing with BPMN+TD – an industrial use case

Quang Huan Dong*, ** Birgit Vogel-Heuser*

** Technical University of Munich, Institute of Automation and Information Systems, 85748 Garching near Munich, Germany (e-mail: {huan.dong, vogel-heuser}@tum.de)*

*** Vietnamese-German University, Vietnam*

Abstract: Relocating manufacturing plants is a phenomenon nowadays due to increasing changes in markets, materials, and labour cost. Under business pressure, while transferring the manufacturing systems, some technical compromises might be taken to start the production as soon as possible. A technical shortcut can provide a short-term benefit but may introduce a long-term negative impact. This work reports an industrial use case at a world-leading electronics manufacturer during its plant relocations. The study employs the Business Process Model and Notation (BPMN) to visualize the use case. There, an enlargement of BPMN, namely BPMN+TD, is achieved to assist in modelling TD artifacts.

Keywords: Business Process Modeling, Integration of Knowledge/Competence in Enterprise Modelling Framework, PLM Systems.

1. INTRODUCTION AND MOTIVATION

In globalization, mass-production companies, such as electrical/electronics (EE) manufacturers, may have their departments in different countries. For example, the R&D department roots in Europe, and the manufacturing sites are located in low-cost countries in Asia. Over time, the manufacturing facilities are often transferred from a higher-cost location to another lower-cost one due to cost optimization and entrance to a lower-cost market segment. There, time pressure during transferring manufacturing systems is high, since management wants to start the production as soon as possible, because of, e.g., an urgent customer demand. Time pressure is often a common trigger to a rising topic, in which some technical activities are partially done or skipped. In industry, one can find similar terms of the compromise on technical activities as "workaround", "shortcut", or "exposure". The Technical Debt (TD) concept describes a circumstance in which a technical compromise is taken to gain a short-term benefit (e.g., to meet a deadline) but may cause a long-term negative impact (e.g., cause higher maintenance effort) (Li et al., 2015). TD always relates to costs, unfortunately, the research on these compromises at EE discipline is still limited (Besker et al., 2017), especially at international companies (Dong and Vogel-Heuser, 2018).

Business Process Model and Notation (BPMN), defined in OMG (2011), is widely used for describing the business process, which is a simple knowledge sharing method. Modelling is an essential activity since it is the first activity in the cycle. Furthermore, errors found near the beginning of the cycle are cheaper. However, BPMN and its available extensions, recently surveyed by Zarour et al. (2019), do not address on modelling scenario of a technical shortcut yet.

According to Panetto et al. (2019), three aspects should be considered to assist management decision-support in manufacturing: modelling, industrial engineering, and management. This study proposes an extension to BPMN, the de facto standard for business process modelling, to allow annotating TD to improve processes and, in particular, TD-related activities or decisions later. The main contributions are: (1) selected results of a TD survey on the cooperation of design departments and manufacturing sites in international companies, which are locating or relocating EE manufacturing; (2) BPMN adaptation to model selected TD aspects. The first contribution provides an up-to-date case from an industrial company, presenting the pros and cons of a typical technical shortcut during its plant relocations. The second one assists a better visualization for the use case. Hence, it supports technicians in demonstrating the case to the management team; thus, managers can use the findings in decision-making processes, such as prioritization and repayment for the technical shortcuts.

2. BACKGROUND ON BPMN AND TECHNICAL DEBT IN ELECTRONICS MANUFACTURING

This section highlights the current state of the related work on modelling with BPMN and the research of TD. Related work on other annotations is out of this paper's scope. At first, BPMN and its extensions are shortly introduced, followed by an overview of TD arise in electronics manufacturing.

2.1 BPMN and Its Extensions BPMN+TD

BPMN allows extensibility with domain-specific concepts to enrich the models. Zarour et al. (2019) recently conducted a systematic literature review and reported over 50 BPMN extensions tailored for various domains. Among the surveyed extensions, there are two works relating to the manufacturing field. First, Abouzid and Saidi (2019) introduced the timing of each task in the manufacturing process. Second, Polderdijk et al. (2018) described human physical risks (e.g. heavy lifting)

10.1016/j.ifacol.2021.08.108

in manufacturing processes. However, none of the two work above and other surveyed extensions focused on presenting technical compromises.

2.2 Technical Debt in Electronics Manufacturing

Coined by Cunningham (1993), the TD concept recently gained large attention in the software engineering domain (Li et al., 2015). The TD concept can be applied to the manufacturing domain (Dong et al., 2018). Besker et al. (2017) studied at one company developing production systems. They reported that TD causes significant additional effort. The authors indicated the importance of conducting a further study at mechanical or EE engineering since their study focused only on the software discipline. So far, only a few works have studied TD relating to EE engineering. Dong and Vogel-Heuser (2018) reported two TD use cases at the machine and plant manufacturers. The work did not focus on EE discipline. Waltersdorfer et al. (2020) reported some experiences with TD management in production systems engineering. They surveyed TD relating to data exchange between four data providers (i.e., mechanical, electrical, fluidics, and software engineers) and two data consumers (i.e., project manager and simulation engineer). The work addressed on development phase; thus, the exchange with data consumers in the subsequent phases (e.g., manufacturing) was not in focus yet. Vogel-Heuser and Bi (2020) reported new TD types in mechatronic products. However, TD modelling was not in the scope, and the research focused on companies in the German-speaking region only.

In conclusion, the relevant research on TD of the EE manufacturing domain is still insufficient to the best of our knowledge, especially in a global context. This work investigates TD at international EE manufacturers. A selected TD use case is modelled with BPMN to support TD visualization. Thus, this first attempt on BPMN and TD integration can allow the knowledge regarding technical shortcuts to be described in a process model.

3. STEPS OF THE RESEARCH

This section described two phases conducted in this qualitative case study. At first, a preparation phase studied obstacles in current EE engineering to build up TD-related questions for the survey. The second phase (i.e., execution and reporting) started with a survey from multiple sources to allow triangulation, followed by modelling and reporting steps.

At the preparation phase, we conducted three initial interviews with people from three German industrial companies, which operate on different continents, to study some industrial TD examples. The EE discipline participants included a product manager of control devices, a project manager of process measurement devices, and an analysis specialist. An interview guideline was iteratively developed and tested during this phase. The interview guideline started with an introduction about the TD concept, followed by TD-related questions. The guideline was revised with different sets of questions and practical terms for different scenarios. As an interviewee could be a manager or a specialist in the design department or manufacturing site, one might have a different perspective on a technical compromise. The challenges during international cooperation were also in the focus of the interviews.

In the second phase, the main interviews were performed to collect a large number of TD examples, followed by feedback loops. In total, there were 13 semi-structured interviews conducted with people from design departments and manufacturing sites in five international companies (i.e., companies with global manufacturing facilities). There were seven interviews from Germany by Webex and six face-to-face onsite interviews in Singapore. The roles of the experts varied from manager to specialist. The interviews last 40 minutes on average. Each interview partner received an individual summary of use cases and TD types found. The experts were able to check whether the cases were formulated correctly and gave feedback. After feedback on the individual summary received, each company's results were integrated and sent to participants in that company. If needed, more investigations were performed with the interviewees to study the cases carefully (e.g., feedback loops on an individual or company interview summary). We found an interesting case regarding negotiations between departments on different continents (reported in the next section, Section 4). The study employs BPMN to enable a better visualization for the use case. There, an enlargement of BPMN is proposed to model the TD artefacts, namely BPMN+TD.

4. A DRAFT ENLARGEMENT OF BPMN TO MODEL TECHNICAL DEBT ARTEFACTS

This section describes two main contributions of the work: (1) an industrial TD use case identified in global EE manufacturing, (2) enlargement of BPMN, its applicability on the TD use case, as well as a prototypical tool. The case was collected at one of the world-leading EE manufacturers of industrial sensors. There were two interviews from Germany by Webex and three face-to-face onsite interviews in Singapore at this company. The roles of interviewers included management (two senior managers and one department head) and specialist (two technological professionals). Due to confidentiality agreements with the participated companies, only abstracted and anonymized results are presented.

4.1 A Technical Debt Use Case in Global Electronics Manufacturing

According to the experts of the EE manufacturer, a product's life cycle often starts with innovation performed in Europe, then the product gets commodity with competition in Singapore, followed by competition in other Asian countries. The process is due to cost, which the manufacturing facilities move from a high-wage country to a lower-wage country. The original work task instruction was short and, in most cases, in a European language, e.g., only few pages with computer drawings or text saying "insert a screw here, put the button there". In Europe, the instruction needed for execution can be condensed and rely a lot on the operators' experience.

When the instructions come to Singapore, "put the button there" and "put the screw here" in words and 1D or 2D drawing are not enough. The local operators would like to see at a specific angle what kind of screw to screw it in a position that fits. Thus, pictures of the facilities are taken during transferring

activities, and then the Europeans have to verify. After a while, the work task instruction length has significantly increased (e.g. multiple times longer than the original European version).

Later, the plant is transferred to another lower-cost Asian country (e.g., Vietnam). Unfortunately, English is not the primary language there. Due to differences in understanding and differences in languages, the same instruction was reworked again. The instruction was expanded in great details and photos for third world production because clarity is needed for execution. At the same time, it is also part of cultural differences where people do not want to do the wrong thing. In a third world production, unskilled operators require a lot of support in terms of photo explanation since a slight misunderstanding can lead to an unexpected manner.

A short-term gain is achieved as the reworked instruction can be used by local operators, who have only to follow the given sections. However, for each model transferred or updated, its work task instruction must go through the above procedure since the original European version is insufficient. Therefore, the company struggles to maintain large instruction for many models since it has become a picture-orientated production instruction. Hence, the additional cost for these activities can be foreseen.

This use case introduces two novel types of TD. First, the *documentation or work processing scheme does not fit the staff's knowledge need* (e.g. in-between Europe, Singapore, and Indonesia/Vietnam). As this TD occurred during the industrial engineering phase, it is classified as an ***Industrial engineering TD***. Second, there *lacks intercultural communication on production instructions for manufacturing*. This TD is classified as a ***Production planning TD within the Industrial engineering TD*** group. For a TD recovery method, the industrial engineers are working on global standardization of production instructions so that it is written and understood in the same manner to avoid extra work and possibly resulting in inconsistencies.

In summary, the use case illustrates an industrial incident throughout a long lifetime. An accumulation of TD occurs, which indicates a high increase of effort to dissolve the technical compromise (e.g., due to shortcuts building upon each other, new expansion based on the shortcut). It is suggested that a collaboration of industrial engineers from headquarter and subsidiaries would be enhanced (e.g. gather feedback and jointly document work instruction) to avoid the reported TD.

4.2 Proposal of BPMN+TD Extension

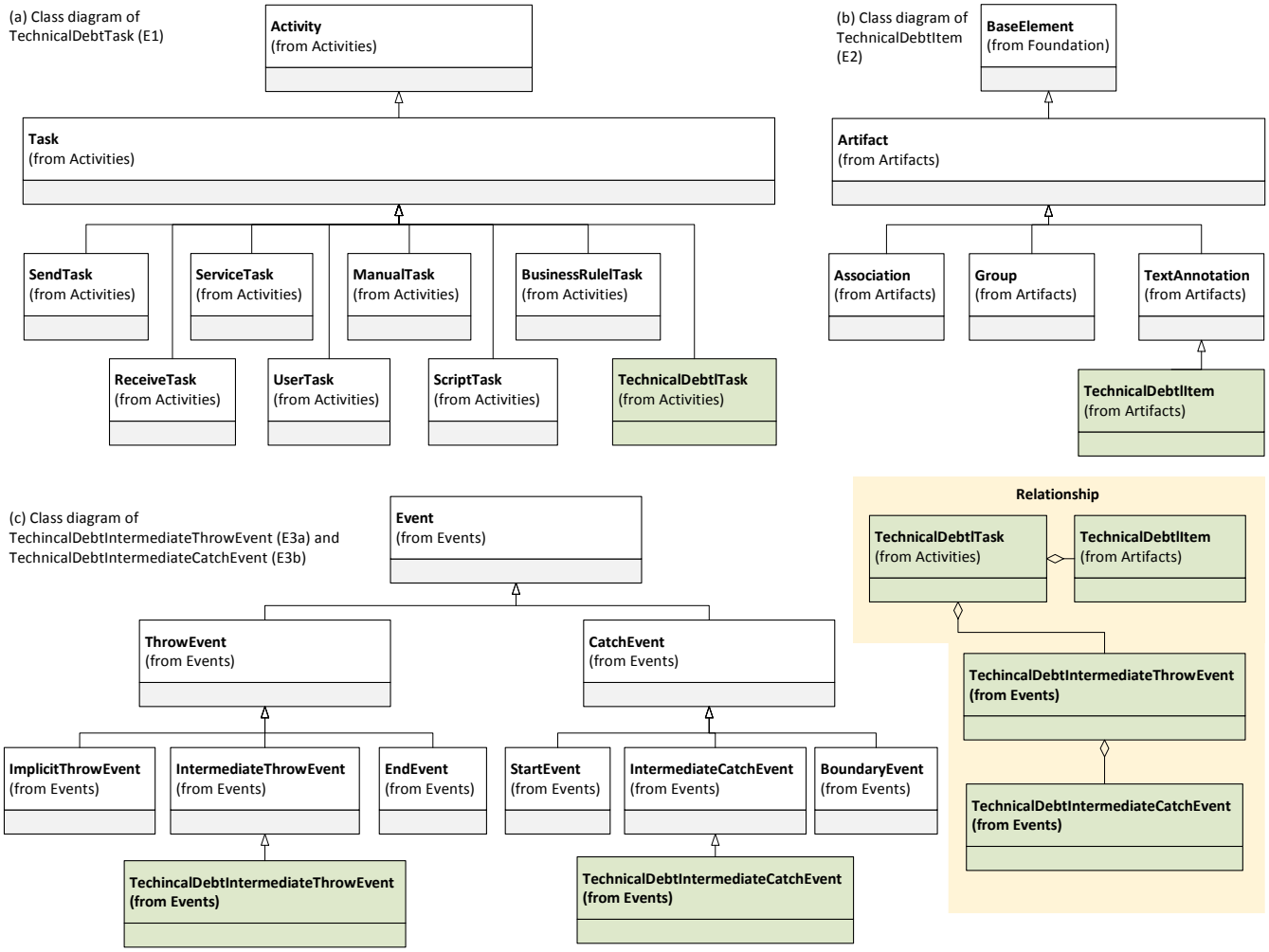


Fig. 1. Enlargement of BPMN to model technical debt (extended classes are highlighted in green)

In order to model the reported use case, an enlargement of BPMN to model TD artefacts is proposed. The BPMN extension composes: modification of the activity box for TD task (E1), TD item (E2), two TD events (E3a, E3b), two association arrows (E4a, E4b), and engineering prototype (E5). An excerpt of UML class diagrams of E1, E2, E3a and E3b is depicted in Fig. 1. The elements are visualized in Fig. 2. The E[notation] in Fig. 1 and Fig. 2 denotes the same element. Firstly, *TD task* supports to model the action of taking a compromise. BPMN defines several tasks to represent different behaviours. For instance, "SendTask" (cp. Fig. 1a) is designed to send a message. The task list is extended with a new task type, namely TechnicalDebtTask. Secondly, *TD item* supports to describe the causes (i.e. type of TD) and the consequences of taking a TD. Thirdly, two *TD events* support

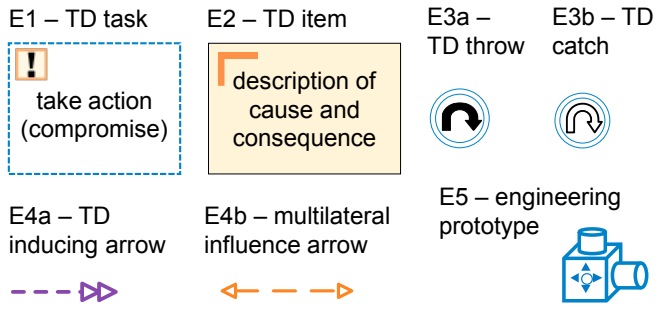


Fig. 2. Proposed additional elements to present TD in BPMN+TD

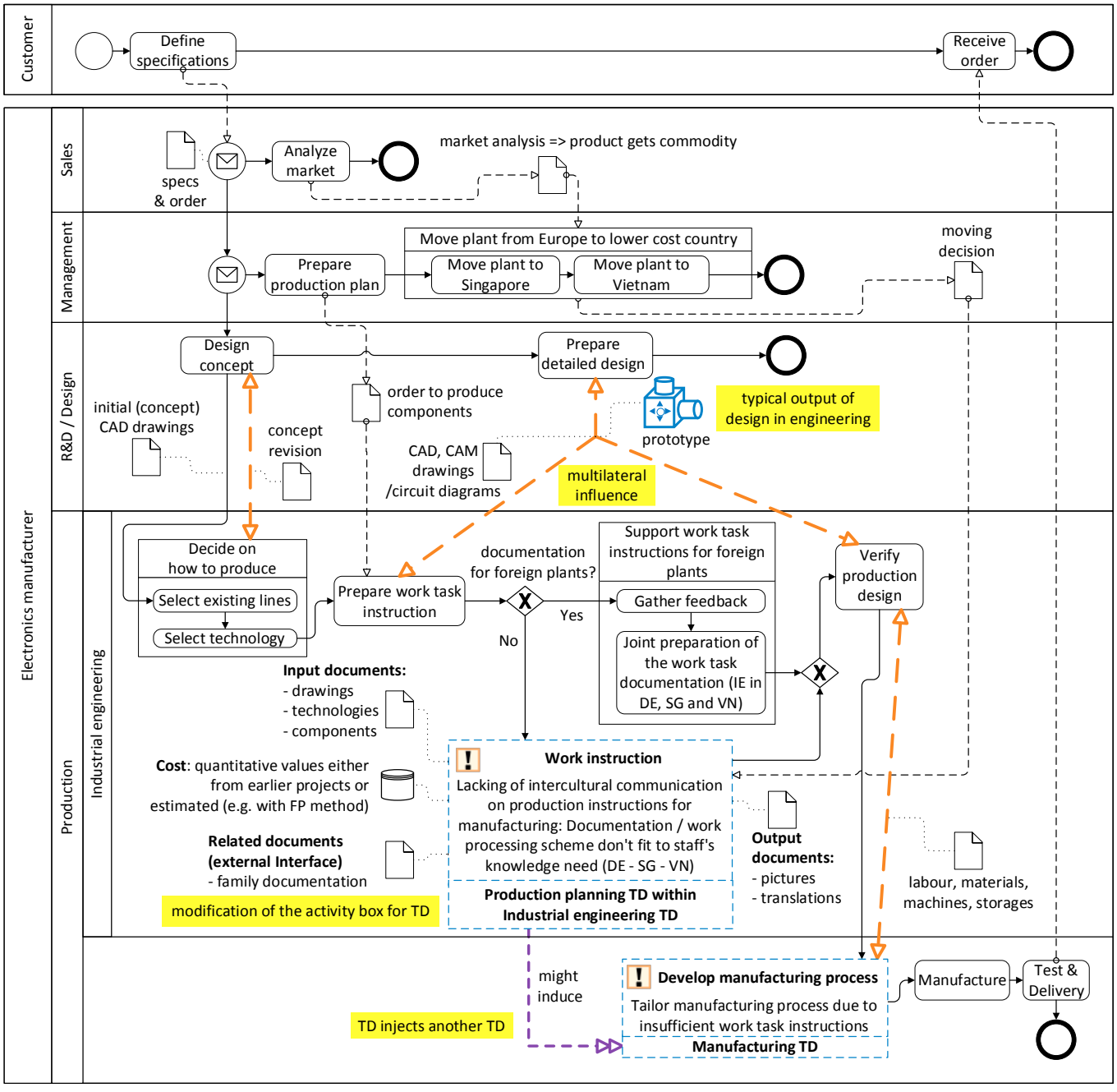


Fig. 3. Modelling a TD use case with BPMN+TD

attaching a TD task with a TD recovery process. TD throw event (cp. Fig. 1-c) generates a TD event and continues the current task. TD catch event listens for a TD event and starts the TD recovery process in parallel. Forth, two supplementary association arrows are introduced: *TD inducing arrow* (E4a), representing one TD injecting another TD, and *multilateral influence arrow* (E4b), representing an association of multiple activities. Fifth, since the prototype is a typical design output in engineering besides the documents, the *engineering prototype* (E5) is introduced.

4.3 Applicability of BPMN+TD on the Technical Debt Use Case

A preliminary model of the reported use case is shown in Fig. 3. It is noteworthy that the activities in Fig. 3 were from the EE manufacturer, in which the use case was studied. The steps are formulated by analyzing the interview data. These steps and their orders can be slightly different in the business processes of other companies. The specification begins at the customer, then comes to sales, management, design, and finally to industrial engineers. The activities of each role are arranged in lanes. Design and industrial engineering will work closely on the drawings as well as the prototype. An application of TD task (E1) is presented at the "*Work instruction*" activity, which is lacking in the exchange with local production personnel. It should be noted that in Fig. 3, the causes and consequences of TD are still represented in the same box of TD task since it is just the preliminary model from the initial analysis. Since the TD task box is too large, later, in the prototypical implementation part (reported in section 4.4), the causes and consequences are presented in a separated element (TD item) for better visualization. A suggestion for TD recovery is described in the box "*Support work task instructions for foreign plants*". An application of multilateral arrow is presented in three activities "*Prepare detailed design*", "*Prepare work task instruction*", and "*Production design verification*". At these three activities, besides documents, prototypes are often exchanged.

At this EE manufacturer, in the box "*Decide how to produce*", "*select existing lines*" activity is often performed first, then new solutions are explored and introduced into the manufacturing process. In a particular case, first, production is started with a common glue, which bonds materials together. The bonding process might take hours; then, research is taken to apply a better glue reducing the process to minutes. However, in general, the order of the two activities, "*select technology*" and "*select existing lines*", may depend on several factors such as complexity of the product or planned cost. On the one hand, a complex product might require a special technology, which influences how the lines are selected or newly setup. There, "*select technology*" can be performed first, then "*select existing lines*" is done afterwards. On the other hand, some technologies are too expensive, and the current budget cannot cover them; thus, "*selecting existing lines*" could have more influences. There, "*select existing lines*" can be performed first, then "*select technology*" is executed afterwards.

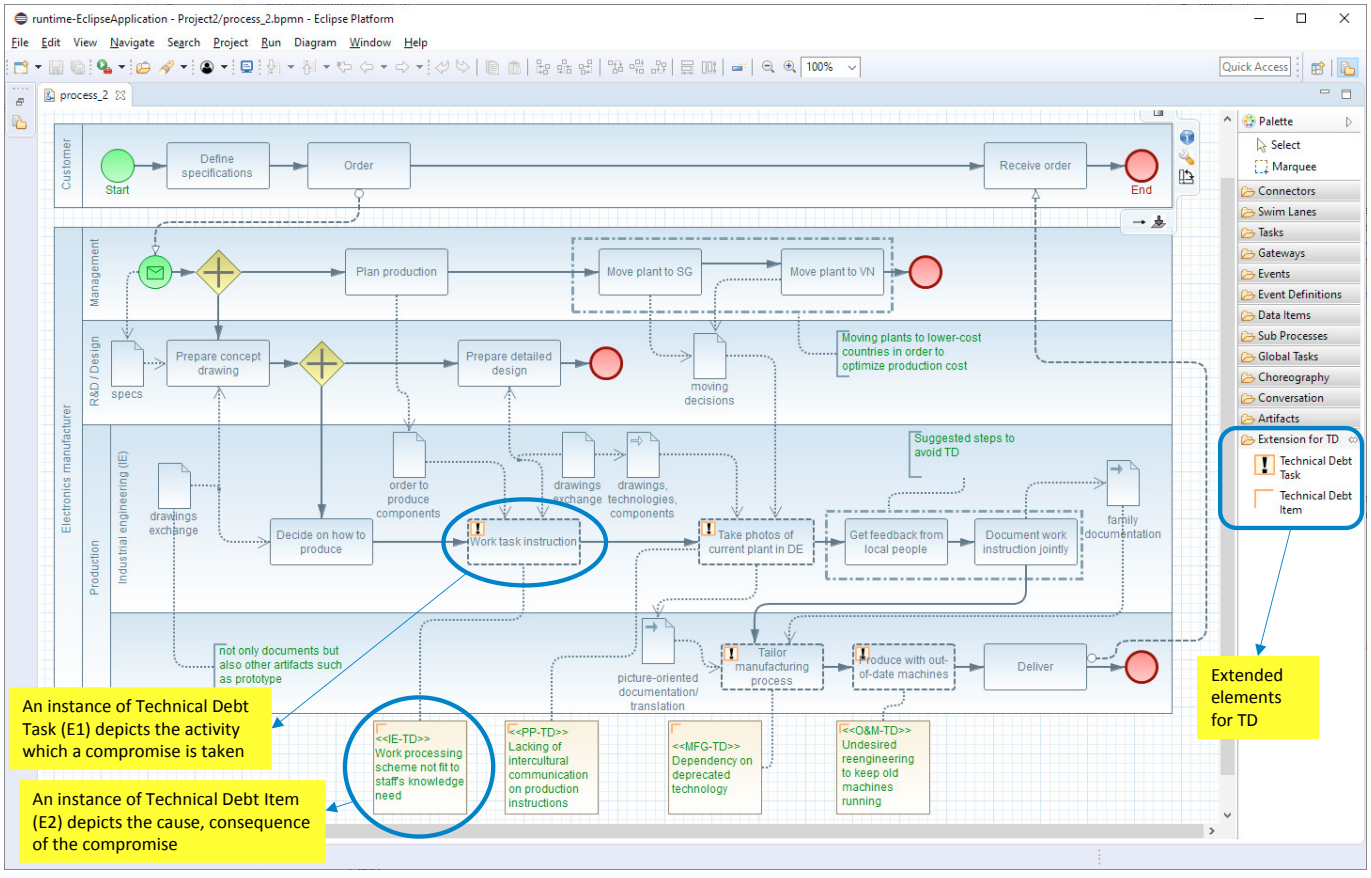


Fig. 4. Screenshot of the prototypical BPMN+TD extension in Eclipse

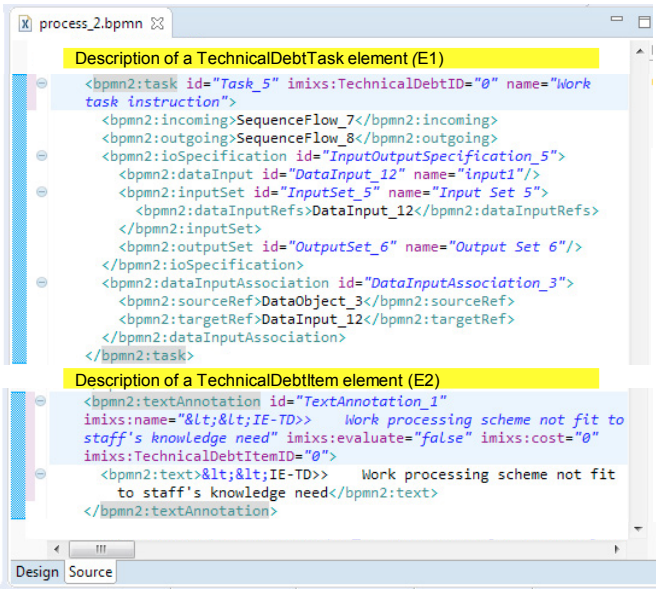


Fig. 5. Excerpt of automatically generated-XML by BPMN+TD plugin on two elements TechnicalDebtTask E1 and TechnicalDebtItem E2 highlighted in Fig. 4

4.4 Prototypical Implementation of BPMN+TD

A prototypical implementation of TD adaptation for BPMN is being developed (cp. Fig. 4). Two graphical notations are supported, TechnicalDebtTask (E1) and TechnicalDebtItem (E2) in the first version. The tool is a plugin BPMN+TD for Eclipse, an extension of an existing Eclipse plugin BPMN2 Modeler (2020). BPMN+TD enhances the BPMN2 Modeler with proposed notations to model TD use cases. An example description of two TechnicalDebtTask and TechnicalDebtItem elements in Fig. 4 is illustrated in Fig. 5. TechnicalDebtTask inherits from Task element, and TechnicalDebtItem inherits from TextAnnotation (cp. Fig. 1-a,b). The description is in XML format, generated automatically by the tool. One can import the generated XML file from the BPMN2 Modeler into a process execution engine. There, instances of the process model can be created, and the documentation activities can be assigned to appropriate staff, thus ensuring drawings are updated by responsible staff or department. Hence, it can track changes at each manufacturing site to analyze the accumulation of the changes, and the information inserted into the BPMN+TD model can prevent knowledge loss in case the related staff already left the company.

5. CONCLUSION AND OUTLOOK

In this paper, we reported an up-to-date TD use case regarding work task instruction, which had been occurring in the industrial engineering phase at a global electronics manufacturer during its plant relocations. An initial enlargement of BPMN for TD was proposed in order to visualize the TD use case. The paper illustrated how a technical compromise and its propagation could be described in a process model. The model depicted the involvement of different roles and linked the action leading to TD as well as its causes and consequences. A preliminary version of BPMN+TD prototypical implementation was proposed with two enlarged graphical notations supported (E1 and E2). The study involved three aspects suggested in Panetto et al. (2019) (i.e. modelling, industrial engineering, and management); thus, the results could assist management decision-support in manufacturing. There, the model could support the technicians in demonstrating the TD case to the management team to start prioritizing repayment for the technical shortcuts.

Further study is planned to assess the findings at global enterprises operating in other industries, such as automotive or machine and tool manufacturing. Future work could incorporate different aspects of TD (e.g., notation for TD interest or TD removal activities linked to TD events) to build up comprehensive TD notations for BPMN. Besides EE disciplines, the application of BPMN+TD could expand into directly related fields such as mechanical engineering.

ACKNOWLEDGEMENTS

The authors thank all companies that have participated in the survey for their contributions. Also, we thank Felix Ocker and Fandi Bi for their support in this research. Finally, we thank the Vietnam International Education Development and Vietnamese-German University for granting Quang Huan Dong a scholarship under Project 911 - "Training lecturers of Doctor's Degree for universities and colleges for the 2010-2020 period" (Decision No. 771/QD-BGDDT dated 14/03/2017).

REFERENCES

Abouzid, I., & Saidi, R. (2019). Proposal of BPMN extensions for modelling manufacturing processes, in: *ICOA 2019*, pp. 1–6.

Besker, T., Martini, A., Bosch, J., & Tichy, M. (2017). An investigation of technical debt in automatic production systems, in: *MTD 2017*, pp. 1–7.

BPMN2 Modeler (2020). Eclipse BPMN2 Modeler [online]. Available at: <https://www.eclipse.org/bpmn2-modeler/> [Accessed 26 November 2020].

Cunningham, W. (1993). The WyCash Portfolio Management System. *ACM SIGPLAN OOPS Messenger*, 4(2), pp. 29–30.

Dong, Q. H., Vogel-Heuser, B. (2018). Cross-Disciplinary and Cross-Life-Cycle-Phase Technical Debt in Automated Production Systems: Two Industrial Case Studies and a Survey. *IFAC-PapersOnLine*, 51(11), 1192–1199.

Li, Z., Avgeriou, P., Liang, P. (2015). A Systematic Mapping Study on Technical Debt and Its Management. *J. Syst. Softw.*, 101, pp. 193–220.

Object Management Group (OMG): Business Process Model and Notation (BPMN), Version 2.0. http://www.omg.org/spec/BPMN/2.0/PDF/ (2011)

Panetto, H., Iung, B., Ivanov, D., Weichhart, G., Wang, X. (2019). Challenges for the Cyber-Physical Manufacturing Enterprises of the Future. *Annu Rev Control*, 47, 200–213.

Polderdijk, M., Vanderfeesten, I., Erasmus, J., Traganos, K., Bosch, T., van Rhijn, G., Fahland, D. (2017). A Visualization of Human Physical Risks in Manufacturing Processes Using BPMN, in: *BPM 2017*, pp. 732–743.

Vogel-Heuser, B., Bi, F. (2020). Interdisciplinary effects of technical debt in companies with mechatronic products – a qualitative study. *J. Syst. Softw.*, 171, 110809.

Waltersdorfer, L., Rinker, F., Biffl, S., Kathrein, L. (2020). Experiences with technical debt and management strategies in production systems engineering, in: *TechDebt 2020*, pp. 41-50.

Zarour, K., Benmerzoug, D., Guermouche, N., & Drira, K. (2019). A systematic literature review on BPMN extensions. *Bus. Process Manag. J.*, 26(6), pp. 1473-1503.

Paper4

IFAC-PapersOnLine 54/4 (2021), pp. 212-219.
https://doi.org/10.1016/j.ifacol.2021.10.036

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-4 (2021) 212–219

Modelling Industrial Technical Compromises in Production Systems with Causal Loop Diagrams

Quang Huan Dong*, ** and Birgit Vogel-Heuser*

** Technical University of Munich, Institute of Automation and Information Systems, 85748 Garching near Munich, Germany (e-mail: {huan.dong, vogel-heuser}@tum.de)*

*** Vietnamese-German University, Vietnam*

Abstract: Automated production systems (aPS) have a long lifetime, which requires significant maintenance cost. During the commissioning phase of aPS, which often occurs at customer sites, the on-site engineers are always under high pressure because the customers would like to start the production at the earliest. There, some compromises, so-called technical debt (TD), might be chosen. A technical compromise taken in the commissioning phase can provide a short-term benefit, but could yield a long-term negative impact on the maintenance phase. This study aims to apply Causal Loop Diagram (CLD) to model technical compromises in the aPS domain. A concept, namely CLD^{+TD}, is proposed. Application examples of CLD^{+TD} on some technical compromises collected from an industrial company working in automation are reported. A loop dominance analysis on a CLD^{+TD} model is performed to derive the circular trend of technical compromise over time; thus, relevant maintenance activities could be planned accordingly. In addition, challenges leading to the technical compromises addressed in the presented use cases can support practitioners in developing appropriate solutions to manage the reported technical compromises in the commissioning phase.

Keywords: production systems, technical debt, causal loop diagram, case study.

1. INTRODUCTION AND MOTIVATION

Nowadays, automated production systems (aPS) are widely used to produce goods in many industry sectors such as automotive, electronics, general air technology, and medicine. As a subset of mechatronic systems, aPS include closely interwoven interdisciplinary components (i.e. mechanical, electrical/electronics, and software components). aPS are often highly adapted based on specific customer requirements (Vogel-Heuser et al., 2015). Due to the large size and/or weight of the aPS modules, it is often impossible to integrate and verify the whole system in the development phase. The first time that many modules fully integrated is in the commissioning phase, which occurs at a customer site. As the commissioning phase is done on-site, there is always high business pressure on the engineers, as the customer wants to start the production as soon as possible. This pressure is often a common trigger that could drive technical tasks to be only partially done or skipped entirely. One can find terms such as "workaround" or "shortcut" describing these technical compromises in the industry.

Coined by Cunningham (1993), the Technical Debt (TD) metaphor describes a context in which a technical shortcut is chosen despite better knowledge. TD is a well-known metaphor in the classical software engineering domain. The TD metaphor has just been introduced into the aPS domain in recent years (Vogel-Heuser et al., 2015). Since aPS have a long operation time, up to decades (Vogel-Heuser et al., 2015), the impacts from TD in aPS may be large. TD always relates to cost; unfortunately, the research on these compromises in the mechanical or electrical disciplines is still limited (Besker et al., 2017) (Dong et al., 2019).

Modelling is one of the most critical engineering activities since a model is a simple knowledge-sharing method. Also, errors found in the early engineering phases are cheaper to resolve. Unfortunately, the methodology to support TD modelling activity is not well addressed in the literature (Li et al., 2015). Recently, Franco (2020) proposed an application of Causal Loop Diagrams (CLD) to model relationships of software maintenance factors in the classical software engineering domain. The model included two factors of TD: Maintainability violations (TD principal) and TD interest amount.

TD in software source code could be automatically detected and presented using code analysis tools. Thus, the visual analysis could be performed, and maintenance activities could be planned to manage code TD timely. However, mechanical or electrical TD is more complicated since it is not easy to detect TD in mechanical or electrical artefacts automatically. In addition, once identified, the TD is often not well documented due to limited methodology to model TD, especially TD in mechanical and electrical engineering. Therefore, the analysis on mechanical or electrical TD is hampered.

CLD allows modelling and analysing interdependencies and connections in between the related factors. Thus, a CLD adaption for TD visualisation may help understand and then monitor the effects, amount as well as the interest of TD. Following the idea in Franco (2020), this study attempts to extend CLD to model TD artefacts in the aPS domain,

10.1016/j.ifacol.2021.10.036

focussing on TD relating to mechanical and electrical disciplines. Also, a loop dominance analysis, which is another method in system dynamics, is performed on the model to derive circular trends of TD over time. Thus, the main contributions of this work are: (1) CLD^{+TD} concept, (2) application examples of CLD^{+TD} on TD use cases identified at an industrial automation company, and (3) challenges leading to TD addressed from the presented TD use cases.

2. BACKGROUND ON CAUSAL LOOP DIAGRAM AND TECHNICAL DEBT

First, this section provides a brief introduction of the System Dynamics approach and its methods, such as CLD or Reference Behaviour Patterns (RBP). Second, the related work on TD and its modelling in the domains of classical software engineering as well as aPS are highlighted.

2.1 Causal Loop Diagram

System Dynamics modelling, an application of Systems Theory, was introduced in Forrester (1961) under the management science research theme, which aims to develop a basis for dealing with management problems. The System Dynamics approach focuses on explaining problematic behaviour and how it emerged, using the "feedback" concept. Causal Loop Diagrams, one of this approach's methods, supports practitioners to model the inner structures of a system, i.e., causal relations between the factors (i.e., variables) involved, as feedback loops. With CLD, it is possible to conceptualise and construct the circular connections and feedbacks in a problem. The diagram consists of nodes representing factors and their relationships represented by arrows. The relationships can be positive or negative, respectively introducing two-loop behaviours, i.e., reinforcing loop or balancing loop. System Theory can enable transferring knowledge between isolated disciplines as well as predict the behaviour of a problem, according to Haraldsson (2004). To predict the trend of a factor, one can perform a loop dominance analysis to derive RBP on a CLD model. The application of System Dynamics modelling was expanded to study problems in many other areas such as sustainability of population growth and global warming, according to (Haraldsson, 2004).

2.2 Technical Debt

According to Li et al. (2015), TD "is a metaphor reflecting technical compromises that can yield short-term benefit but may hurt the long-term health of a software system". The TD topic recently gained considerable attention in the classical software engineering domain. Findings in Martini et al. (2016) expressed that if the amount of TD is not well managed, extensive refactoring or a redesign would be required to add new features into the system later. If the interest of TD is small, the effect of TD is negligible; if interest is large, the system becomes unsustainable (Digkas et al., 2021). The systematic mapping study in Li et al., (2015) indicated that the TD representation or documentation activity received the least attention among TD management activities, as there was only four percent of the surveyed studies relating to this activity. The TD representation or documentation activity is an essential step since it can provide useful information for other TD management activities such as TD measurement or visualisation trends of TD. Thus, the amount of TD can be monitored (i.e., TD monitoring activity). Based on Ground Theory, Martini et al. (2015) proposed a crisis point model and evaluated it with interviews. The case study proposed a more detailed model, which includes several variables of constantly accumulated ATD, phases, and kinds of refactoring. From these initial models, they inferred trends of TD when applying different refactoring strategies. However, those models were just partially confirmed. Recently, Franco (2020) has used CLD to model relationships of software maintenance factors based on Lehman's (1996) laws. Two TD factors (TD principal and TD interest amount) were included in the model. The study showed that different software maintenance strategies could impact the amount of TD and TD interest. Although the benefit of TD was considered, the TD benefit factor was not presented in the model. Modelling TD artefacts in a specific use case was not addressed, as the research focus was on the strategic level.

In the aPS domain, Vogel-Heuser et al. (2015) reported some TD examples in mechanical and electrical disciplines. However, the reported TD items were only from the authors' personal experience or a laboratory demonstrator. Besker et al. (2017) conducted a study at one company developing production systems and reported that TD causes significant additional efforts. The authors indicated the importance of further research in mechanical or electrical engineering since their study focused on the software discipline only. Dong et al. (2018) reported one mechanical TD example at the design phase at a machine manufacturer and one software TD example in a plant manufacturer's commissioning phase. Vogel-Heuser et al. (2020) reported several industrial TD examples in mechatronic products. There were two electrical TD taken before the commissioning phase: (1) "liquid test was not performed before the commissioning" and (2) "faulty sensor selection and lack of initial commissioning in the factory". Waltersdorfer et al., (2020) reported some TD examples relating to the engineering data exchange between four data providers (i.e., mechanical, electrical, fluidics, and software engineers) and two data consumers (i.e., project managers and simulation engineers). The work focused on the development phase; thus, the exchange with data consumers in subsequent phases (e.g., on-site engineers at commissioning, customers, etc.) was not considered. Unfortunately, none of the above work focused on modelling TD artefacts in the aPS domain.

In summary, to the best of our knowledge, there is a research gap regarding the modelling of TD artefacts in the aPS domain, and the work on industrial TD use cases relating to mechanical and electrical disciplines in the aPS domain is also limited. Based on Systems Theory and the idea in Franco (2020) in the software engineering domain, this work employs CLD to model some TD use cases collected at a company in the automation domain and derives circular trends of TD from a loop dominance analysis. In this first study, we focus on TD related to commissioning when there is great business pressure; thus, the occurrence probability of this TD kind is high.

3. STEPS OF THE RESEARCH

Four main steps were conducted in this study. In the first step, TD use cases were collected by conducting semi-structured interviews with experts from three German companies working in the automation domain. The interviewees included: (P1) a specialist at a very large (over 10000 employees) automotive manufacturer M1, (P2) a product manager at a large field level device manufacturer M2 (250-9999 employees), and (P3, P4) two senior managers at a large field level device manufacturer M3 (same range as M2). P1 and P2 were involved in electrical and software disciplines. P3 and P4 were involved in the mechanical domain. Thus, the different areas of expertise of these four interviewees are sufficient for a first study. All interviews were audio-recorded and transcribed for further analysis. In the second step, a CLD^{+TD} concept was proposed, based on the idea in Franco (2020). In the third step, the concept was applied to some TD use cases. A loop dominance analysis was also applied on the simplest CLD^{+TD} model example. In the final step, an evaluation of CLD^{+TD} usefulness was conducted using an online survey.

4. CLD^{+TD} CONCEPT TO MODEL TECHNICAL DEBT ARTEFACTS

This section first proposes an extension of CLD to model TD artefacts, namely CLD^{+TD}, followed by a loop dominance analysis on the CLD^{+TD} model to derive RBP, which shows circular trends of a TD.

4.1 Proposal of a CLD^{+TD} extension and modelling Technical Debt artefacts with CLD^{+TD}

Based on the definition of TD in Li et al. (2015), three main factors of the TD metaphor are formulated: (1) TD benefit (i.e., "short-term benefit"), (2) TD interest (i.e., "hurt the long-term health of a software system"), and (3) TD. Also, one TD could induce another TD, for instance, due to interwoven interdisciplinary relationships in the aPS domain. Thus, the CLD^{+TD} extension includes: (1) a new factor, namely TD benefit, which was not presented in Franco's model, (2) and a supplementary association arrow, namely "TD inducing arrow". The graphical notations to model TD artefacts in detail are illustrated in Fig. 1: (1) three modifications of the factor box for TD, TD interest and TD benefit, (2) and TD inducing arrow.

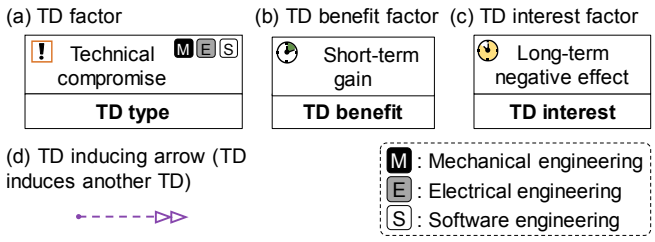


Fig. 1. Proposed additional elements to present TD in CLD^{+TD}.

A CLD model of the three TD factors is illustrated in Fig. 2. The reinforcing loop indicates a growth of TD benefit and an increase in TD taken. The balancing loop indicates an increase in TD interest resulting in a decrease in TD taken. As TD interest may occur in the long-term (i.e., TD interest's effect may not be showing for a long while), a delay is added into the balancing loop.

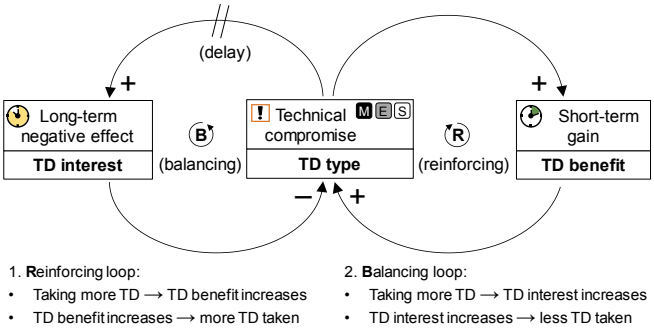


Fig. 2. CLD^{+TD} concept – Modelling Technical Debt with Causal Loop Diagrams.

4.2 Deriving circular trend of Technical Debt from a loop dominance analysis on CLD^{+TD} model

Based on the idea in Haraldsson (2004), a loop dominance analysis is performed on the CLD^{+TD} model to derive BRP. A loop dominance describes which part of the feedback in a CLD is the strongest (most active) at a given time. Five steps of this analysis are illustrated in Fig. 3. The phases are presented in Fig. 3-(a1) to (e1), and Fig. 3-(a2) to (e2) illustrate how the amount of TD is influenced. The first step presents the initial phase (idle stage). The four following steps are highlighted with thick arrows showing which loop is experiencing loop dominance behaviour. The second step presents the first loop dominance, which is the TD growth phase. The TD is allowed to grow since the TD benefit increases. At the third step, the feedback starts, and the loop dominance becomes the decline of TD due to the increment of TD interest. The amount of TD interest continues for some time until the growth of TD causes the system to be unsustainable. At the fourth step, the amount of TD decreases at the decline phase. It could be explained that due to TD interest increases, the maintenance cost increases; therefore, less TD is taken, or some TD recovery activities are conducted. At the fifth step, according to RBP, it is possible to express the levels when the amount TD stabilises qualitatively, and when it occurs, the growth phase of the TD starts again.

5. APPLICABILITY OF CLD^{+TD} ON INDUSTRIAL TECHNICAL DEBT USE CASES

First, this section reports preliminary research results by presenting application examples of CLD^{+TD} on three industrial TD use cases obtained from the interview with the experts P3 and P4. There are three tiers involved in these use cases. Tier 1 (customer) represents a company producing the final products (e.g., car wash machine) for the consumers. Tier 2 (machine or plant manufacturer/constructor) represents a company producing the machines or establishing the plants for Tier 1. Tier 3 represents a company producing the devices or platforms (e.g., sensors) for Tier 2. Tier 3 may also deliver devices to Tier 1. The three use cases show the strong influences from the purchasing department of Tier 1 and Tier 2 on the decisions to take TD in the automation, influencing all tiers participating. Among three use cases, the CLD^{+TD} model of use case 1 is the simplest one. Thus, firstly, the loop dominance analysis results on that model are presented as an

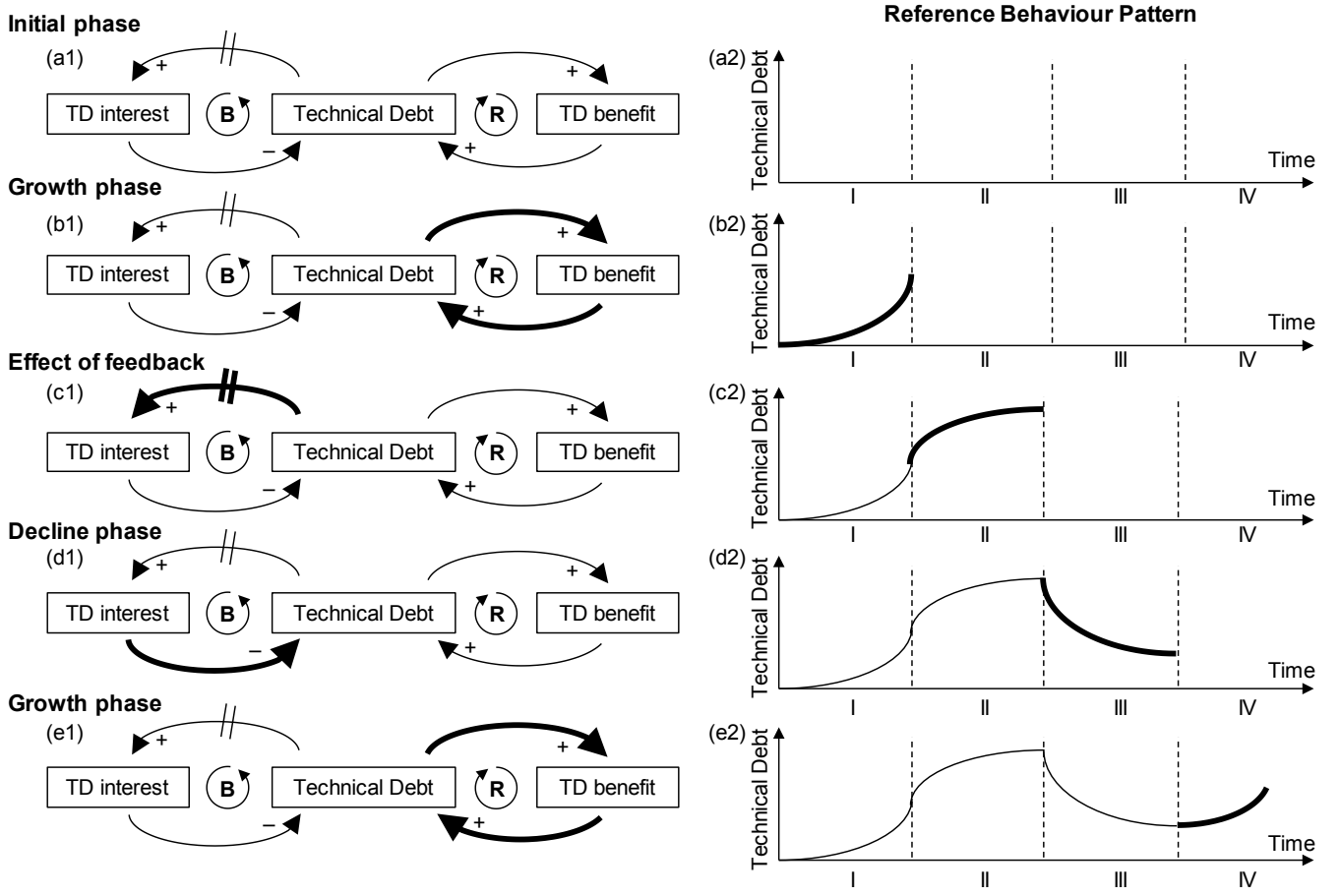


Fig. 3. Circular trend of Technical Debt derived from a loop dominance analysis on CLD^{+TD} model based on the idea in Haraldsson (2004); thick arrows indicate reinforcing or balancing effect of other factors on the Technical Debt factor.

illustration. Due to confidentiality agreements with the participants, only abstracted results are available. Secondly, the section reports a synthesised model and some challenges leading to TD addressed in the use cases. Thirdly, an initial result of the online survey is presented. Fourth, the result and improvement ideas are discussed. Finally, the section ends with a description of the study limitations.

5.1 Application example on Technical Debt use case 1 – selecting sub-optimal manufacturing process to meet short-term financial goals

The experts reported a sub-optimal manufacturing process that was based on a low-cost solution. A system or process is considered sub-optimal if it does not satisfy certain desired qualities. For example, the heat of one device could be reused in another with a more expensive solution. The initial investment could enable significant energetic savings for the customer (Tier 1) in the following years. Unfortunately, the decision-makers in the customer's purchasing department do typically not stay longer than two years in the company, according to the expert. Thus, because of short-term financial goals, the optimal solution was abandoned by those decision-makers. Therefore, the plant was operating at a sub-optimal energy utilisation level.

The TD factor was "sub-optimal energy utilisation level", classified as an Architectural TD as it occurred at the strategic level. The TD is influenced by the "customer financial goal" factor, representing a purchase department's decision or an insufficient budget reserved. The TD benefit factor identified from this use case was "short-term cost-saving", and the TD interest was "long-term energy cost". Application of the CLD^{+TD} on the TD artefacts identified in this use case is illustrated Fig. 4.

Following the steps presented in section 4.2, a loop dominance analysis on the CLD^{+TD} model of the TD use case (Fig. 4-a) is performed. The TD use case model includes one reinforcing loop and one balancing loop, which is adapted from the concept model (Fig. 3-e1). Thus, the trend of TD's "sub-optimal energy utilisation level" would be quite similar to Fig. 3-e2. The result is depicted in Fig. 4-b. However, a faster increment at phase I in Fig. 4-b is illustrated, as there is a contribution of the "customer financial goal" factor.

5.2 Application example on Technical Debt use case 2 – installing sub-optimal equipment due to expectation gap

According to the experts, there was a sub-optimal equipment installation between the tiers. Tier 2 extended its service to Africa but did not communicate the local requirements to Tier 3. Thus, Tier 3 delivered the same devices to the "new" African plant of Tier 1 as to the European plants. Unfortunately, low-

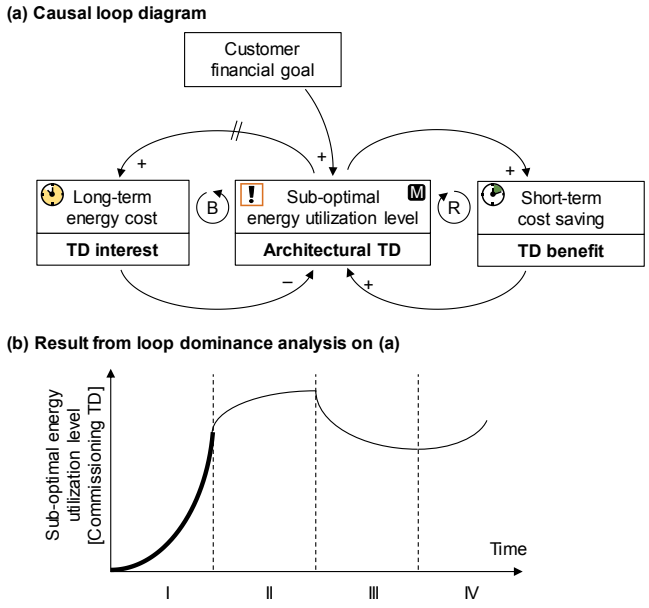


Fig. 4. Modelling and loop dominance analysis of case 1.

quality parts were chosen at the African plant, and there was an improper installation for connected components. Moreover, different environmental conditions such as higher acidity and fluid flows in the process caused faster wear and tear issues on the devices. Therefore, the devices malfunctioned, and a high maintenance rate was caused. Unfortunately, the failure was not apparent, so the experts required a long time for troubleshooting, and the plant had to be shut down during the investigation. The insufficient specifications caused high additional costs, damage, and delay for the tiers.

In this use case, the identified TD factor was "inappropriate selected components", classified as a Commissioning TD. The low-quality accessories were chosen due to "quick and cheap" reason, according to the interviewee. Thus, the TD is influenced by the "financial goal" factor, which relates to decisions from the purchasing departments at Tiers 1 and 2. In addition, the TD is also influenced by the "time pressure" factor. It should be noted that there are similar use cases where inappropriate components are selected due to lack of necessary documentation/guideline and insufficient supervision from Tier 3 to Tier 2 in the commissioning phase at Tier 1. Thus, the TD might also be influenced by the "documentation quality" factor. These inadequate aspects are connected to the "expectation gap" factor, as Tier 3 often expected sufficient competencies of Tier 1 and Tier 2, and the delegation of responsibility, and vice versa. The TD benefit factor identified from this use case was "short-term cost-saving", and the TD interest factor was "maintenance rate". Application of the CLD^{+TD} on the TD artefacts identified in this use case is illustrated in Fig. 5.

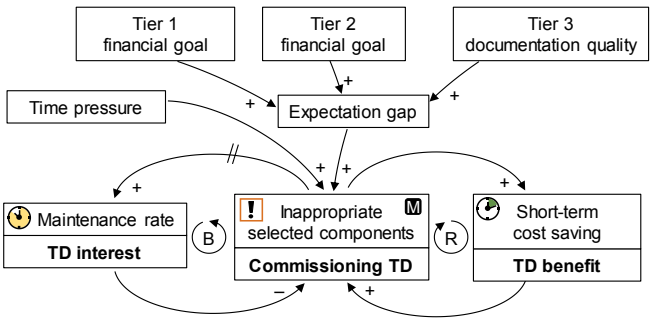


Fig. 5. Modelling TD use case 2.

5.3 *Application example on Technical Debt use case 3 – complicated software change instead of adding expensive hardware*

The experts described a case that there was a sub-optimal design for a process plant. Due to a cheaper way of processing raw materials (chemicals), a slow method of measuring temperature changes was selected (measuring temperature with pressure), which required additional implementation in software and electrical engineering. Due to the delay in the temperature measurement, an overheating of the raw materials was induced, and a part of the desired end products was destroyed. This induced a bad product quality. An optimal solution would require more expensive equipment and would be based on the physical relation of converting the materials. The process would be quicker as direct temperature control would not be required. Moreover, the control strategy in electrical or software engineering would be more simple. However, more than 50% of the customers tend to choose the sub-optimal solution, according to the interviewees.

The first TD factor was "choosing sub-optimal material processing strategy", classified as a Design TD. The design TD is influenced by the "customer financial goal" factor, which represents a decision from the purchasing department or an insufficient budget reserved. It could be explained that the terms of the purchasing decision-makers (typically several years) are often much shorter than the plant lifetime (typically several decades). The design TD induced a commissioning TD, namely "additional implementation to cover design TD". The TD benefit factor identified from this use case was "short-term cost-saving". The first TD interest factor was "control strategy complexity" as additional control mechanism need to be implemented in software and electrical engineering. The second TD interest was "manufacturing production inefficiency", representing the slow processing and bad product quality which may cause the additional rejects. An application of CLD^{+TD} on the TD artefacts in the TD use case is illustrated in Fig. 6.

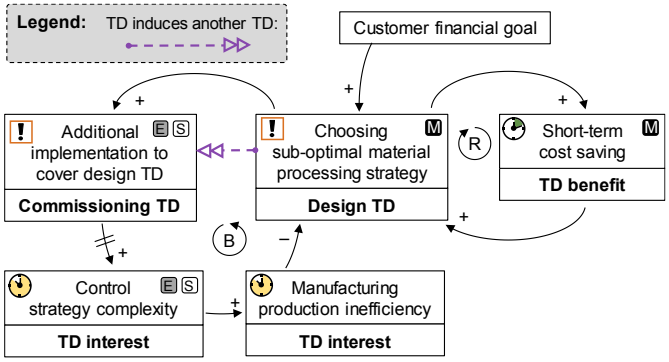


Fig. 6. Modelling TD use case 3.

5.4 A Synthesised Model and Challenges Leading to Technical Debt in Presented Use Cases

To provide an overview, a synthesised model of the three presented CLD^{+TD} models is depicted in Fig. 7. Some factors are appearing in multiple use cases. For instance, the Short-term cost-saving factor appears in all three use cases. Thus, the frequency of occurrence of some factors is noted (cp. Fig. 7).

Several challenges were leading to TD addressed. The first challenge identified in use case 1 is that future savings were not counted in cost calculation, which leads to a low efficient operation. In use case 2, three challenges were addressed: (1) a lack of communication in term of preparing necessary documentation/guidelines and supervision from Tier 3 to Tier 2 (constructor) in the commissioning phase at Tier 1, (2) a gap in expectation from Tier 3 to Tier 2 and Tier 1 on technical competences (the ability to select the right technologies/parts) as well as delegating responsibilities regarding system's reliability and compatibility, and (3) impacts from decisions from the purchasing department at Tier 1 and Tier 2 to meet a "short-term financial goal". Two challenges were identified in use case 3: (1) impacts from interdisciplinary decisions (e.g., cross-disciplinary TD induced from mechanical engineering to electrical and software engineering), which are difficult to be identified and monitored, and (2) financial decisions due to short-term objectives.

There might be a relationship between the three use cases. Due to the decision to select a low resource-efficient solution to meet a short-term financial goal (use case 1), sub-optimal equipment might be installed (use case 2), then complicated software change instead of adding expensive hardware (use case 3). Thus, three use cases might relate to the purchase departments' decisions (represented by the "financial goal" factors).

5.5 An Evaluation Using an Online Survey

To check the usability of the CLD^{+TD} approach, an evaluation was conducted with the interviewees using an online survey (TUMAIS, 2021). Although the use cases and models were implemented into the questionnaire, this survey only focused on evaluating the approach's usability, not on confirming the use cases. There are some differences between the use cases and models in the survey and the ones published in the paper. So far, two responses were received from P1 and P3 interviewees. The survey contained two parts: ten evaluation questions on the approach and application examples and three questions to check whether the experts had understood the models. Overall, the ratings of interviewee P1 are better compared to the ratings of P3 (cp. Fig. 8).

5.6 Discussion and Improvement Ideas

Hereafter, Q[number] denotes a question. The "quite good" score 4 (moderately agree) at Q1 shows that P3 could understand the CLD^{+TD} concept well. However, from Q2 to Q10, the ratings of P3 were gradually decreased, compared to the score at Q1 (cp. Fig. 8-a).

At Q2, both P1 and P3 rated a "neutral" score 3. This may be caused by the differences in the concept model in Q1 and the concrete model in Q2. The concept had one TD type box and two loops, but the concrete case had two TD type boxes and two loops. Moreover, the TD-inducing phenomenon symbol was just briefly introduced (TD-inducing arrow), thus, might hinder the understanding. Thus, Q2 introduced more information compared to Q1. Therefore, a participant might have problems with Q2 if they did not understand Q1 well enough. Nevertheless, P1 rated Q2 better than Q1 since Q1 showed an abstract model, and Q2 presented a concrete case.

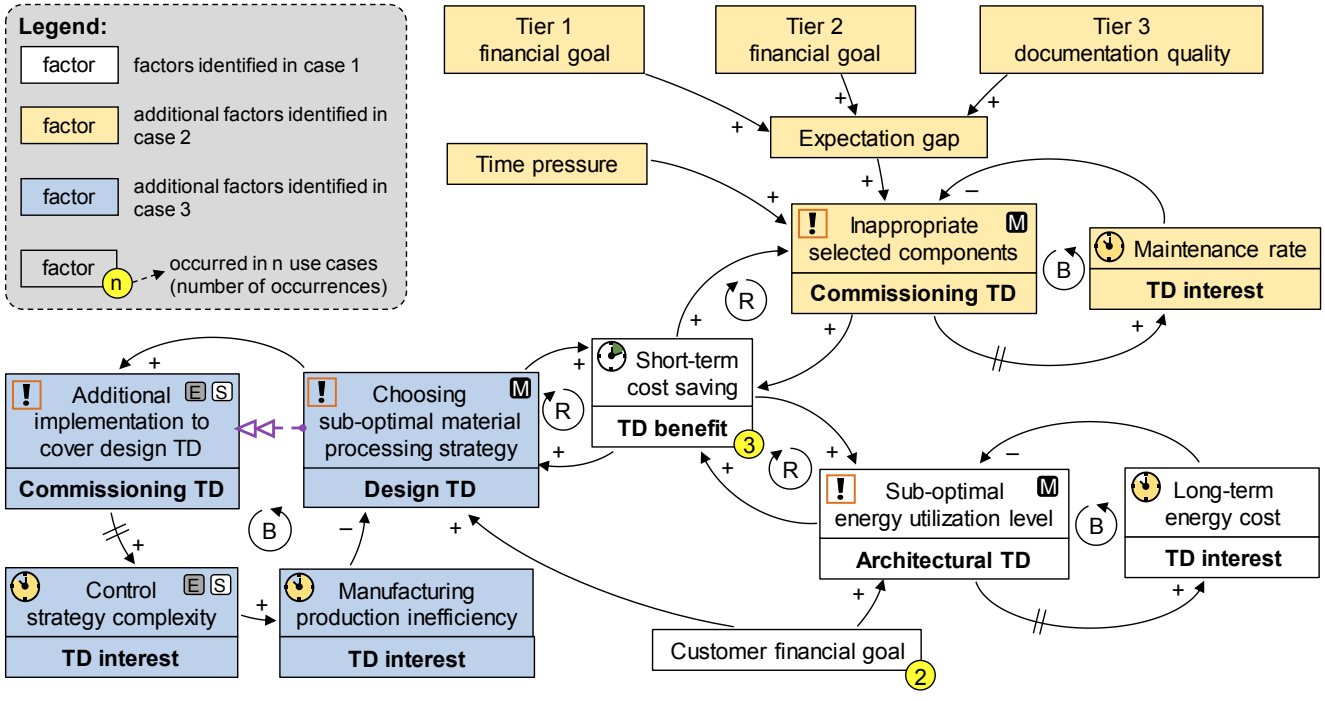
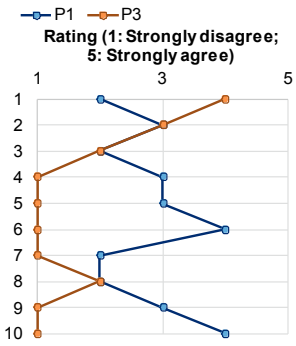


Fig. 7. Synthesized model of three presented technical debt use cases.

(a) Usability of CLD+TD



Question	Related figure
I think that CLD+TD concept is easy to understand	Fig. 2
I think that CLD+TD provides a sufficient visualization of this Technical Debt use case 1	Fig. 6
I think that CLD+TD provides a sufficient visualization of multiple Technical Debt use cases	Fig. 7
I think that CLD+TD can support me to present Technical Debt to different stakeholders in a uniform manner	
I think that the concept (circular trend of Technical Debt) is easy to understand	Fig. 3
I think that the trend graph provides a sufficient overview on the amount of a TD overtime	Fig. 6, Fig. 4-b
I think that loop dominance analysis on CLD+TD can support Technical Debt management activities …	
I think that CLD+TD is easy to visualize my own TD use cases	
I think that I would use CLD+TD to visualize my own TD cases frequently	
I would imagine that most people would learn to use CLD+TD very quickly	

(b) Understanding of CLD+TD

	P1	P3	Corrected answer	Question
11	False	True	False	The main purpose of CLD+TD is to identify Technical Debt.
12	True	False	True	The main purpose of CLD+TD is to document Technical Debt.
13	True	False	True	CLD+TD can represent a Technical Debt related to multiple fields such as mechanical, electrical and software engineering

Fig. 8. Result of the online survey.

At Q3, both P1 and P3 rated 2, which is a "not so good" score. Introducing the complex model of three use cases in Q3 might hinder the understanding since not all individual models of use cases were separately introduced in previous questions. Thus, the ratings at Q3 went down, perhaps due to the steep increment of complexity in Q3 with multiple cases and possibly the lack of understanding on previous question Q2 with one case. Therefore, an overview was not sufficiently provided because of the model complexity.

Regarding Q4 to Q10, since P3 might not understand the previous questions well, it might be even harder to understand the next questions, thus, rated "low" scores. As three answers of P3 at the understating questions were incorrect (cp. Fig. 8-b, Q11-Q13), the "low" ratings of P3 may relate to understanding. Thus, a follow-up discussion was conducted with P3 to identify the reason for the low ratings. The feedback was that there was too much information in the visualisation (models) which caused P3 to lose track. The different scores from P1 and P3 may relate to their different roles, as P1 was a specialist and P3 was a manager. P1's ratings were better as the models might be more understandable for a technical end-user. The low rating at Q7 from P1 may be due to the too general trend of TD (similar to Fig. 4-b), which is not sufficient for supporting TD management activities yet.

A lesson learnt was that it could ease the understanding of the participants if different levels of detail on the CLD^{+TD} models are provided. To provide this feature, different views on the visualisation could be introduced. For example, a group of factors could be encapsulated as a sub-factor and could be expanded in a detailed view or collapsed in a more abstract view. This feature could improve the neutral ratings at Q2. This feature could also support the practitioners in presenting TD to different stakeholders in a uniform manner, thus addressing the not-so-good ratings at Q4.

Another lesson learnt was that a large model – too much information – should not be suddenly introduced. A step-by-step explanation could address the low ratings at Q3. To address the "not so good" ratings at Q5, Q8, Q9, and Q10, more materials such as videos or a workshop could help. More detailed examples and a simulation of some models could improve the low rating at Q7 in future evaluation.

The hierarchy relationship between factors of use cases could be presented in a 3D view. CLD^{+TD} models can be displayed at different layers (e.g., strategic to implementation). This view could enhance the visualisation of the synthesised model. A novel prototypical hierarchy view is illustrated in Fig. 9. There, the root cause and hierarchy influence of the factors and the use cases can be visualised. Further improvement could denote the involved/responsible personnel at each factor or layer.

Compared with former approaches, firstly, the visualised trend of TD from the CLD^{+TD} approach confirms the crisis point

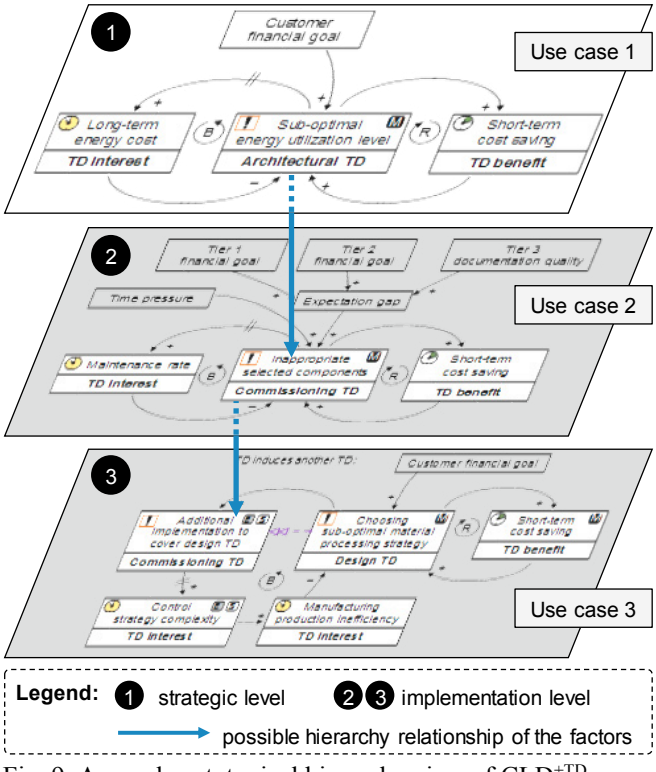


Fig. 9. A novel prototypical hierarchy view of CLD^{+TD}.

model in Martini et al. (2015). However, the approaches, focus and results of CLD^{+TD} and Martini et al. are a bit different. Martini et al. based on Grounded theory and focused on the trend of TD when different refactoring strategies were applied. CLD^{+TD} bases on Systems theory and aims at presenting the circular trend of TD with the concept of feedback. Secondly, CLD^{+TD} incorporates the TD benefit factor, which was not presented in the model of Franco (2020). While Franco mainly focused on the strategic level, CLD^{+TD} considers implementation, strategic levels and cross-disciplinary aspect.

5.7 Limitations of the Study

There is a limitation as the selection of discussed TD use cases was based on convenience sampling. Despite the limitation, it is worth noting that all interviewees were professionals with significant experience in the automation domain. Another limitation is that the study was conducted with companies in two industries (automotive and industrial automation). Hence the results may not be generalised to other industries. Since the study was undertaken only with large and very large companies, the results cannot be generalised to small and medium companies. Nevertheless, the study contributes a concept that was evaluated with three TD use cases from a large organisation, which could provide a decent degree of generalizability.

6. CONCLUSION AND OUTLOOK

This paper reports three Technical Debt use cases at a German industrial company working in the automation domain. A concept, namely CLD^{+TD}, is proposed to model the TD artefacts with Causal Loop Diagrams. Application examples of the CLD^{+TD} concept on the reported TD use cases are evaluated and presented. The CLD^{+TD} models could support practitioners to present TD use cases to different stakeholders in a uniform manner. The initial findings from a loop dominance analysis on one CLD^{+TD} model at the strategic level can help practitioners predict the roughly amount of TD in aPS over the phases, thus planning for extra maintenance activities accordingly. Practitioners in the automation domain should examine occurrences of the reported TD challenges in their companies to develop appropriate solutions to manage the reported TD items.

Future work should perform further evaluation, analyse the proposed model's convenience, and explore the applicability of the approach. To predict the amount of TD, TD benefit, and TD interest more precisely, future work can perform simulation on the presented CLD^{+TD} models at the implementation level using software such as Simantics System Dynamics (2020). Future work can explore the cross-tier TD in the international context.

ACKNOWLEDGEMENTS

The authors thank all companies that have participated in the survey for their contributions. Also, we thank Iris Weiß, Emanuel Trunzer, and Fandi Bi for their support in this research. Finally, we thank the Vietnam International Education Development and Vietnamese-German University for granting Quang Huan Dong a scholarship under Project 911 - "Training lecturers of Doctor's Degree for universities and colleges for the 2010-2020 period" (Decision No. 771/QD-BGDDT dated 14/03/2017).

REFERENCES

Besker, T., Martini, A., Bosch, J., & Tichy, M. (2017). An investigation of technical debt in automatic production systems. *Proceedings of the XP2017 Scientific Workshops on - XP '17*, pp. 1–7.

Cunningham, W. (1993). The WyCash Portfolio Management System. *ACM SIGPLAN OOPS Messenger*, 4(2), pp. 29–30.

Digkas, G., Ampatzoglou, A., Chatzigeorgiou, A., Avgeriou, P., Matei, O., Heb, R. (2021). The Risk of Generating Technical Debt Interest: A Case Study. *SN Computer Science*, 2(1), 12.

Dong, Q. H., Vogel-Heuser, B. (2018). Cross-Disciplinary and Cross-Life-Cycle-Phase Technical Debt in Automated Production Systems: Two Industrial Case Studies and a Survey. *IFAC-PapersOnLine*, 51(11), 1192–1199.

Dong, Q. H., Ocker, F., Vogel-Heuser, B. (2019). Technical Debt as Indicator for Weaknesses in Engineering of Automated Production Systems. *Production Engineering*, 13(3–4), 273–282.

Forrester, J. W. (1961). *Industrial Dynamics*. Productivity Press.

Franco, E. F. (2020). A Dynamical Evaluation Framework for Technical Debt Management in Software Maintenance Process. *PhD thesis*, Universidade de São Paulo and the Università degli Studi di Roma "La Sapienza".

Haraldsson, H. V. (2004). *Introduction to system thinking and causal loop diagrams*. Department of Chemical Engineering, Lund University.

Lehman, M. M. (1996). Laws of Software Evolution Revisited, in: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 108–124.

Li, Z., Avgeriou, P., Liang, P. (2015). A Systematic Mapping Study on Technical Debt and Its Management. *Journal of Systems and Software*, 101, 193–220.

Martini, A., Bosch, J., Chaudron, M. (2015). Investigating Architectural Technical Debt Accumulation and Refactoring over Time: A Multiple-Case Study. *Information and Software Technology*, 67, 237–253.

Martini, A., Bosch, J. (2016). Architectural Technical Debt in Embedded Systems. *INCOSE International Symposium*, 26(1), pp. 1029–1043.

Simantics System Dynamics (2020). Simantics System Dynamics | Open Source Modelling And Simulating Tool For Simantics [online]. Available at: <http://sysdyn.simantics.org/> [Accessed 26 February 2021].

TUMAIS (2021). Survey on Usability of Visualizing Technical Debt with Causal Loop Diagram [online]. Available at: < https://umfrage.ais.mw.tum.de/index.php/178284?lang=en&token=Q7NIL> [Accessed 26 February 2021].

Vogel-Heuser, B., Rösch, S., Martini, A., Tichy, M. (2015). Technical Debt in Automated Production Systems, in: *2015 IEEE 7th Workshop on Managing Technical Debt*, pp. 49–52.

Vogel-Heuser, B., Bi, F. (2020). Interdisciplinary effects of technical debt in companies with mechatronic products – a qualitative study. *Journal of Systems and Software*, 171, 110809.

Waltersdorfer, L., Rinker, F., Kathrein, L., Biffl, S. (2020). Experiences with technical debt and management strategies in production systems engineering, in: *TechDebt 2020*, pp. 41-50.

Paper5

Forschung im Ingenieurwesen/Engineering Research. (*submitted on 18 March 2022*)

Title / Titel

Including validation of process control systems' engineering into the Technical Debt classification / Erweiterung der Klassifikation Technischer Schuld-Typen validierungsnotwendiger Prozessautomatisierungssysteme

Author information

Quang Huan Dong[1, 2, *], Birgit Vogel-Heuser[1,3]

[1] Institute of Automation and Information Systems, Technical University of Munich, Germany

[2] Vietnamese-German University, Vietnam

[3] Core Member of MDSI

[3] Member of MIRMI

[*] corresponding author (email: huan.dong@tum.de)

The authors Q.H. Dong and B. Vogel-Heuser (email: vogel-heuser@tum.de; ORCID: 0000-0003-2785-8819) are with the Technical University of Munich, Institute of Automation and Information Systems, Boltzmannstr. 15, 85748 Garching, Germany.

Abstract / Zusammenfassung

The Technical Debt (TD) concept depicts a situation in which a technical compromise is selected despite a better solution available. The associated risks to TD may be critical if TD is overlooked or not fixed properly. For instance, the product might not be used safely (in the healthcare supplies industry) or be contaminated (in the food & beverage sector) due to a design fault not adequately addressed. This work extends the TD classification for process automation systems that need to undergo a validation process according to GAMP (Good Automated Manufacturing Practice), which is required for pharmaceutical production or food processing, for example. The approach analysed industrial engineering documents to identify such TD types and sub-types. Three selected TD use cases confirmed by industry experts are reported. The presented meta-analysis approach on engineering documents can be employed as a TD identification method for such process control systems. The TD classification for process automation was thereby enlarged for a specifically critical type of machine that may harm the health of many humans.

/ Das Konzept der technischen Schuld (Technical Debt, TD) beschreibt eine Situation, in der ein technischer Kompromiss gewählt wird, obwohl eine bessere Lösung verfügbar ist. Die mit TD verbundenen Risiken können kritisch sein, wenn TD übersehen oder nicht richtig behoben wird. Beispielsweise könnte das Produkt aufgrund eines nicht angemessen behobenen Designfehlers nicht sicher verwendet werden (im Medizintechnik) oder mit Schmutz kontaminiert sein (in der Lebensmittel- und Getränkeindustrie). Diese Arbeit erweitert die TD-Klassifizierung für Prozessautomatisierungssysteme, die einem Validierungsprozess gemäß GAMP (Good Automated Manufacturing Practice) unterzogen werden müssen, wie er beispielsweise für die pharmazeutische Produktion oder die Lebensmittelverarbeitung vorgeschrieben ist. Der Ansatz analysierte Dokumente aus der Industrietechnik, um solche TD-Typen und Untertypen zu identifizieren. Es wird über drei ausgewählte, von Industrieexperten bestätigte TD-Anwendungsfälle berichtet. Der vorgestellte Meta-Analyse-Ansatz für technische Dokumente kann als TD-Identifikationsmethode für solche Prozessleitsysteme eingesetzt werden. Die TD-Klassifizierung für die Prozessautomatisierung wurde damit um einen besonders kritischen Maschinentyp erweitert, der die Gesundheit vieler Menschen gefährden kann.

Statements and Declarations

Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

CRediT authorship contribution statement

Conceptualization: Birgit Vogel-Heuser, Quang Huan Dong; Methodology: Birgit Vogel-Heuser, Quang Huan Dong; Formal analysis and investigation: Quang Huan Dong, Birgit Vogel-Heuser; Writing - original draft preparation: Quang Huan Dong, Birgit Vogel-Heuser; Writing - review and editing: Birgit Vogel-Heuser, Quang Huan Dong; Resources: Birgit Vogel-Heuser; Supervision: Birgit Vogel-Heuser.

Acknowledgments

The authors thank the industrial partners for their contributions. Also, we thank Jan Wilch, Fandi Bi, Juliane Fischer and Eva-Maria Neumann for their support in this research. Finally, we thank the Vietnam International Education Development and Vietnamese-German University for granting Quang Huan Dong a scholarship under Project 911 - "Training lecturers of Doctor's Degree for universities and colleges for the 2010-2020 period" (Decision No. 771/QD-BGDDT dated 14/03/2017).

Including validation of process control systems' engineering into the Technical Debt classification

Abstract
The Technical Debt (TD) concept depicts a situation in which a technical compromise is selected despite a better solution available. The associated risks to TD may be critical if TD is overlooked or not fixed properly. For instance, the product might not be used safely (in the healthcare supplies industry) or be contaminated (in the food & beverage sector) due to a design fault not adequately addressed. This work extends the TD classification for process automation systems that need to undergo a validation process according to GAMP (Good Automated Manufacturing Practice), which is required for pharmaceutical production or food processing, for example. The approach analysed industrial engineering documents to identify such TD types and sub-types. Three selected TD use cases confirmed by industry experts are reported. The presented meta-analysis approach on engineering documents can be employed as a TD identification method for such process control systems. The TD classification for process automation was thereby enlarged for a specifically critical type of machine that may harm the health of many humans.

1. Introduction and Motivation

Coined by Cunningham [1] in the classical software engineering domain, Technical Debt (TD) describes a context in which a technical compromise is taken, e.g., delivering low-quality code to meet a short deadline. The technical shortcut can provide a short-term benefit but may cause a long-term negative impact on the system quality or the productivity of engineers [2]. The TD concept can be transferred to the engineering of automated Production Systems (aPS), which are widely used nowadays to assemble products in various fields such as automated packaging, pharmaceutical production or food processing. aPS are a subclass of mechatronic systems. Typically, aPS are developed by engineers from multiple disciplines such as mechanical, electrical or software. Other fields such as hydraulic, pneumatic, sensor, process or drive technology may also be involved in aPS development. There are strong interdependencies between all those disciplines. Analysing TD in the software discipline at one aPS company, Besker et al. [3] suggested further research should study mechanical and electrical fields in addition to software.
During aPS development, technical decisions from various disciplines and phases are validated and described in several documents such as product description, operation manual, risk analysis documentation or technical product requirements. In this work, a meta-analysis is performed to address engineering TD in the aPS domain. The study considers the documents conducted during the development of aPS in the healthcare sector followed V-Model [21] and GAMP 5 [20] (i.e. Good Automated Manufacturing Practice). The examined documents include (1) functional specifications and design specifications at different disciplines (e.g. hardware, software, electrical/electronics) following V-Model, and (2) risk trace matrix and corresponding internal guidelines following GAMP 5.
While TD in the classical software engineering domain can be identified with code analysis, the work on TD in the aPS domain is still limited [5]. Recent work on TD identification in the aPS domain still focuses on software discipline [29]. Thus, a systematic method is lacking to identify TD in other fields in the aPS domain. This work is the first qualitative study on multi-disciplinary engineering documents to study the state-of-the-practice risk analysis to identify and classify engineering TD in the aPS domain. The main contributions of this paper are: (1) a systematical TD identification approach on engineering documents in the aPS domain and (2) enlargement of TD classification with TD sub-types from validation of process control systems' engineering. The work thus yields a sound basis for TD management to improve overall equipment effectiveness (OEE) in the aPS domain as well as enlarges the body of knowledge on TD.
The remainder of the paper is structured as follows: In the next section, state of the art on validation of process control systems of automated manufacturing systems and TD in aPS are outlined. Subsequently, research questions are derived. In Section 3, a meta-analysis approach on engineering documents to identify TD is described. The results are presented in Section 4. The paper concludes with a summary and an outlook on future works.

2. State of the art in production automation TD and validation to identify open research questions

This section presents an overview on the validation of process control systems of automated manufacturing systems, followed by an outline of TD related work, which derives research questions in detail.

2.1. Validation of process control systems of automated manufacturing systems

In the area of qualification for automation technology, the companies have to follow specific guidelines to meet requirements from the authorities such as Food and Drug Administration (FDA) or European Medicines Agency [31] [32] [37]. Good Manufacturing Practices (GMP) should be applied to ensure safety and traceability to comply with the increasingly strict regulations [40]. Alam [35] indicated that "pharmaceutical Process Validation is the most important and recognised parameters of cGMP" (Current Good Manufacturing Practices) from the FDA. Glennon [30] reported that in the validation field of computer control systems, "besides factory acceptance tests, identification of potential problems minimises the risk of costly field corrections".

Smith et al. [33] proposed a math model to support risk analysis on the systems. Chowdary et al. [36] tried to deploy lean tools together with cGMP principles. Examining system life cycle, risk management and system testing, Samson et al. [39] indicated that a life-cycle approach would best suit process systems engineering validation. There are some guidelines such as NAMUR recommendations and worksheets [25] or GAMP. The current generally accepted policy for the validation of computer-aided systems is GAMP 5 (the last version of GAMP), according to [28].

GAMP 5 is a risk-based life-cycle approach to making industry compliant computerised systems; thus, aPS manufacturers can acquire the certificate to market the manufacturing systems. GAMP 5 provides the critical fundamental concept to identify and report risks throughout the life-cycle of systems. For instance, the severity of a fault on product quality is offset against the fault occurrence probability. The calculation determines the risk classes from I to III. The risk class is offset against the likelihood that a fault is identified before damage is caused and results in risk priority categories from Low, Medium to High. The detection or prevention methods are designed to achieve technical improvements and reduce risk take precedence. Consequently, control measures and qualification tests are defined as part of the completed assessment in accordance with company-specific rules. GAMP 5 recommends two rounds of risk priority evaluations for each risk reported, and the risk assessment should be carried out for all changes. Based on GAMP, Neuhaus et al. [27] defined validation procedures for the process control system of a large scale manufacturing plant.

In pharma and food & beverage, Feigenbaum et al. [34] introduced methods to control food migration into GMP. Meyliana et al. [41] proposed a blockchain technique to minimise the circulation of counterfeit drugs. Huysentruyt et al. [43] presented an application of GAMP 5 for AI-based systems. In recent work, Andriani et al. [42] reported a case study of a GAMP application to analyse and prioritise risks in the production process. Suseno et al. [38] emphasised knowledge sharing and GMP performance. Poor exchange of know-how could be a challenge during the implementation of the MES (Manufacturing Execution System) in the food and beverage industry, according to Chen et al. [40].

The risks documented by the GAMP 5 risk-based approach are potential sources for quality improvement. Specifically, the reported risks might reveal technical compromises taken along the engineering process; the discipline initiates the compromises and impacts on the related fields. Examining the compromises could identify new know-hows that could benefit aPS manufacturers and the customers, as TD might impact system quality [2]. Furthermore, a cross-disciplinary TD study is necessary [3]. The following section reviews related work on TD in the automaton domain and its near fields to identify the research gap.

2.2. Technical Debt

Li et al. [2] established a TD classification according to different causes in the classical software engineering domain. Among the TD causes, such as inappropriate architecture, lack of tests, or documentation, the most studied TD was code TD [2]. Avgeriou et al. [4] indicate that TD always relates to cost. Unexpected significant cost overruns in software development projects could occur if failure to monitor TD [6]. TD could have an impact on the product delivery date [7] [8] [9] and the organisation's profitability [10] [11] [12]. However, the engineers sometimes suggest a TD removal plan, but the management is often unwilling to approve the improvement due to the business value focus [13]. Additionally, the staff choosing to initiate TD (e.g., designers) could be different from those dealing with TD (e.g., maintenance engineers) [14] [15].

In the embedded software engineering domain, which is nearer to the aPS domain, Martini et al. [16] examined architecture documentation to study TD. However, the focus was on architectural TD in software only.
In the aPS domain, Besker et al. [18] analysed the work of software engineers at one aPS company in Scandinavia. They reported that the engineers spend significant extra effort due to TD. The reported effort "on average 32% of the development time" on TD was confirmed by managers. The work implies that more investigation should be conducted to understand TD at other disciplines of aPS, such as mechanical or electrical. Vogel-Heuser et al. [19] conducted case studies regarding fault handling in companies developing aPS to increase OEE. However, the work focused on software discipline only. A survey conducted by Dong et al. [5] with the machine and plant manufacturers indicated that TD might introduce unscheduled costs between disciplines and phases. However, TD awareness at these companies is still low. Vogel-Heuser et al. [17] analysed the interdisciplinary effects of TD in companies working with mechatronic products. They found that TD "emerges most frequently in the first three stages of the life cycle" (i.e. Requirements, Architectural and Design). The work was based on the interview method, and examining engineering documents was not in focus.
To the best of the authors' knowledge, no other studies have been undertaken to explore the aPS engineering documents to identify TD in different disciplines of the aPS domain. Thus, this work analyses documents of aPS companies to study engineering TD (**RQ1**). Once identified, the new detected TD are systematically classified (**RQ2**), and the extent of benefit with industrial experts should be evaluated (**RQ3**). Thus, this work aims to fill the research gap to contribute to a potential increase in OEE in the aPS domain.

2.3. Research Questions

Following the idea of Li et al. [2] in the classical software engineering domain, a study should examine typical settings in which TD occurs and which constraints, situations, and individuals or disciplines are involved. As outlined in section 2.1, in some fields such as pharmaceutical production or food processing, aPS need to meet some predefined requirements, which are often validated using GAMP. Following the GAMP principle, quality must be validated at each engineering stage; thus, the study should examine not only decisions or risks documented from engineering phases but also the documents from early/preparation phases. To be precise, engineering TD could be determined by analysing design specifications (**RQ1.1**), as the design is one of the most frequent stages in which TD emerges [17]. Further, the risk trace matrix documents centralised not only the identified risks in the project phases but also the risks during the entire operation; thus, those recorded and traceable risks could be a potential source to identify engineering TD (**RQ1.2**). There, the identified TD might relate to specific engineering phases (**RQ2.1**) or may link to cross-engineering stages (**RQ2.2**), as aPS changes or evolutions are multi-stage and multi-disciplinary [18] [5] [17]. Furthermore, an evaluation with industrial experts is necessary to confirm the identified TD types and the identified use cases. The reported TD cases could provide a soundness proof concerning engineering risks (**RQ3.1**). In addition, the usefulness of reported TD cases could be perceived as a potential improvement for system quality as well as risk management process since uncovered TD might pose risks (**RQ3.2**).

3. Meta-analysis approach on engineering documents

In the beginning, a meta-analysis was conducted with engineering documents provided by a world-leading machine and plant manufacturer which had been adopting V-Model and GAMP 5. The functional specifications followed the V-Model process, and the risk evaluation was in accordance with GAMP 5. The compositions described engineering and risk in electrical, software, and mechanical disciplines. The risk trace matrix document also includes feedback from the customer of the participated company.
The solution mentioned in each risk entry was assessed to check whether there was a suboptimal solution; thus, TD could be detected. Two workflows with different approaches were designed for this study to achieve a TD classification. The two methods supported the researchers with systematically reviewing and identifying TD in significant long documents (more than a hundred pages). On the one hand, the bottom-up approach (cp. Fig. 1a) focused on discrimination between implementation and guidelines/standards. On the other hand, the top-down approach (cp. Fig. 1b) analysed risk entries. Nevertheless, this analysis design could also be reused in other studies which aim at a similar goal.

Regarding approach 1 (cp. Fig. 1a), the first step was the identification of implemented guidelines in specifications of different fields (i.e. functional, hardware, software, electrical/electronics). Some examples of identified aspects were

- the hardware classification followed GAMP 5
- the creation of the software design specification is based on GAMP 5
- The emergency stop and safeguards have been designed and manufactured in compliance with the Directive on Machinery (EC 2006/42)

In parallel, an exploration of potential guidelines on which specifications should be followed was conducted to identify insufficient conformity to the standards or best industry practices. Aspects with no procedure mentioned were studied. This step is an iteration process. Some examples of identified aspects were

- HMI Visual Displays of Machine Status did not mention techniques followed. Some potential candidates are
  - VDI/VDE 3699 Process control using display screens
  - OMAC [26] stack light concept
- Operating Modes did not mention any guidelines or standards followed. The potential candidate is
  - OMAC

The second step was TD identification by studying the identified guidelines and comparing them with the implementations. Also, by examining the "Cause of Malfunction" and "Detection/Prevention" of each risk entry in the risk trace matrix, the documented solution was checked if there was any "sub-optimal" or "quick-fix" which often initiates TD, according to Cunningham [1]. Specifically, the analysis considered some aspects identified in Lueddemann [22] as a potential source of TD. For instance, inadequate reciprocity among hazards, faults, risk control measures, design, implementations, and tests. Other examples are low traceability of residual risks, redundant data sets within technical documentation, or inconsistent use of unique data identifiers across projects. More details of this step are described in Section 4.1.1. The second step was an iterative process. Consequently, the detected TD items were added to the well-known TD classification tree in Li et al. [2]. The detected TD items were further classified following the detailed activities related to the risk assessment process at different phases in Hegde et al. [3]. For instance, some risk related TD items initiated during the design phase were classified into a new sub-type TD named Risk analysis TD within Design TD. Another example is some risk related TD items were organised into a new sub-type named Risk assessment TD within Documentation TD.

In the third step, a snowballing was performed. Recent work and aspects relating to the identified TD (in specifications) were explored to search for potential guidelines. The second step was an iterative process. If more aspects were detected, the process would come back to the second step. Otherwise, the process reached the final goal.

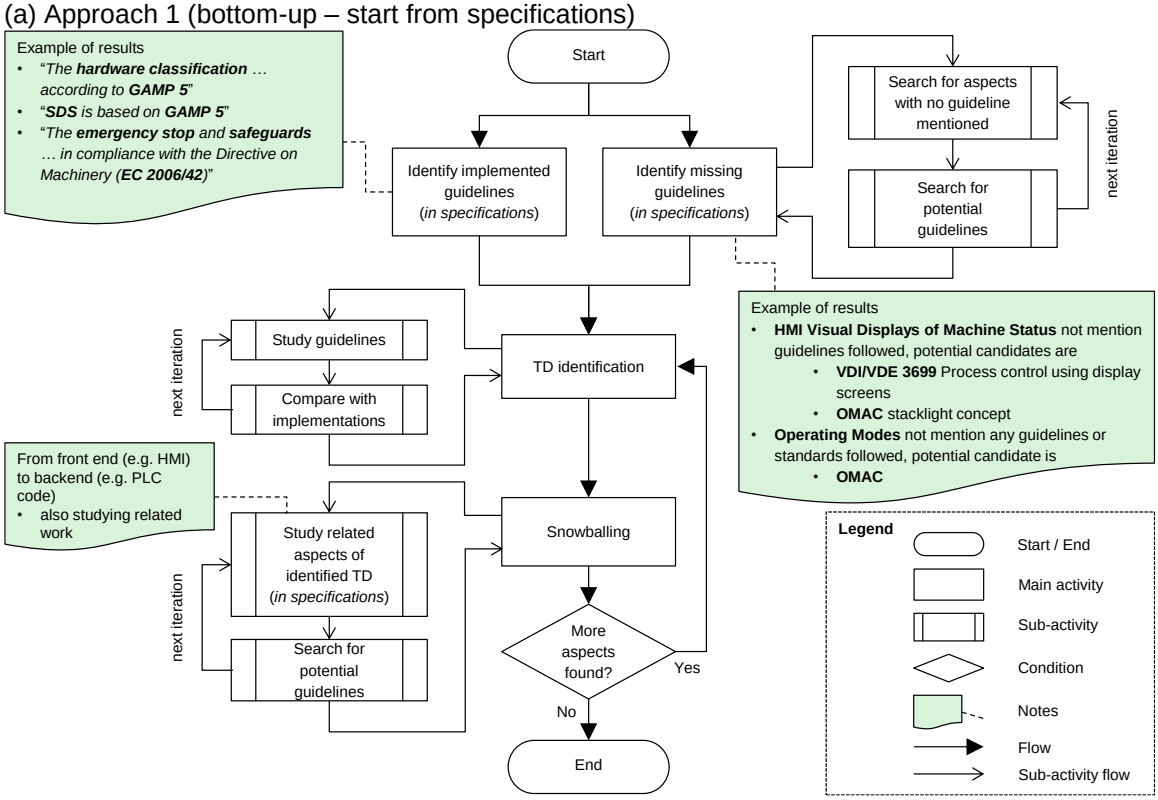
Regarding approach 2 (cp. Fig. 1b), entries in a risk trace matrix were identified in the first step. This step contained two sub-steps: (1) risk aspects were studied, and (2) implemented or potential guidelines were explored. An example of results was "Guard locking does not lock", and a possible procedure was OMAC. This step was an iterative process.

In the second step, TD was identified by two sub-steps: (1) studying guidelines and (2) comparing with implementation. The mentioned guidelines here are specifications from the industrial partner. This step was also an iterative process.

Similar to the bottom-up approach, in the third step, a snowballing was performed. There, related aspects of identified TD (in specifications) were studied to search for potential guidelines. Some related work to the identified aspects was also explored. The second step was an iterative process. If more aspects were detected, the process returned to the second step. Otherwise, the process reached the final goal.

In the following, some examples of the TD identification step are presented, and threats to validity are reported.

(a) Approach 1 (bottom-up – start from specifications)



(b) Approach 2 (top-down – start from risk analysis)

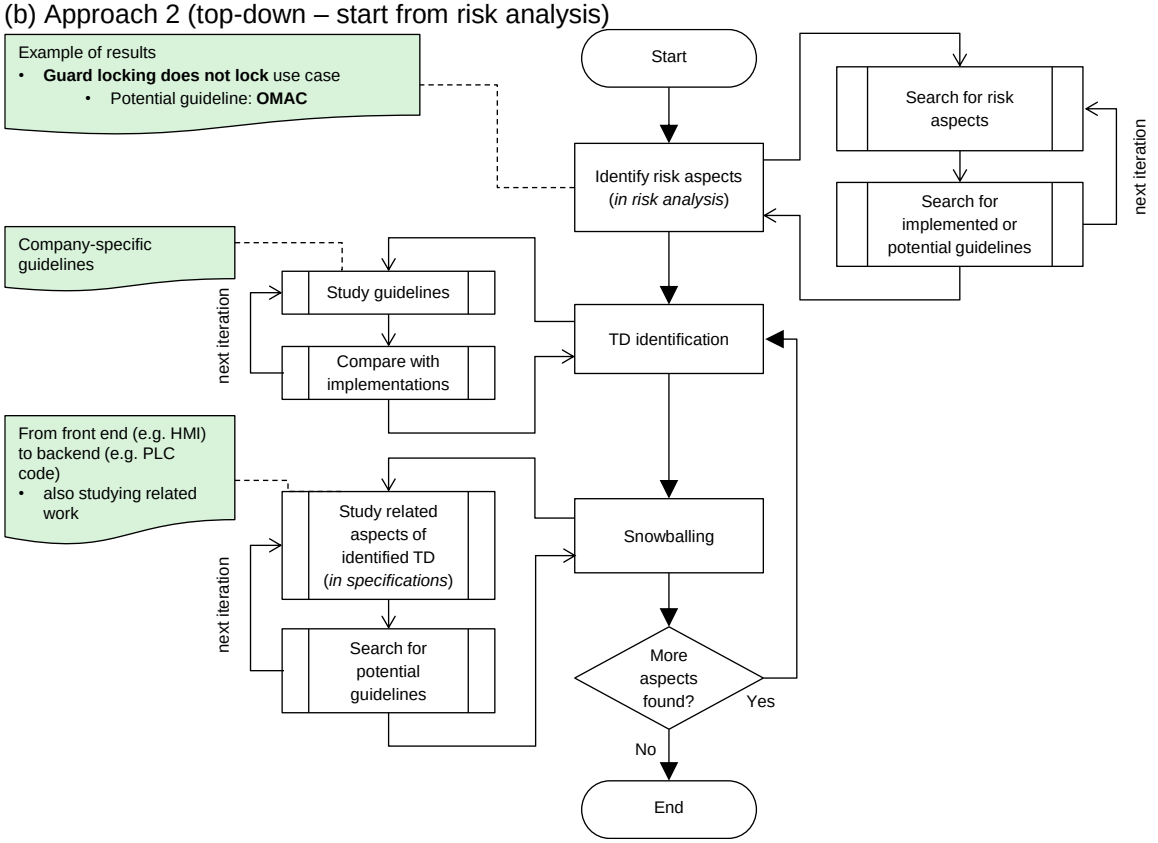


Fig. 1 Meta-analysis approaches on engineering document to identify Technical Debt – (a) bottom-up approach, (b) top-down approach

3.1. Examples of TD Identification Step

This section presents the rationale of TD identification. Three selected examples are used to illustrate how and why a TD was detected. The identified TD items were incrementally denoted as **DocTD.[Number]**. The numbers show the order of the TD detected during the meta-analysis following the approaches presented in Fig. 1.

Regarding the first example, there was a discrepancy between risk assessment guidelines and practice (**DocTD.09**). In the internal risk assessment guideline, the company defined definitions and points to each function in the Severity, Probability and Detection categories. The points were then multiplied to determine risk classes and risk priorities. For example, 1, 3, and 5 points would be given at Severity rating; 2 and 4 evaluation levels would not be used. However, in practice, Severity ratings with 4 points still appeared in the risk trace matrix.

In the second example, there was a discrepancy between practice and GAMP 5 (**DocTD.07**). The internal risk analysis document noted that the risk evaluation process would be in accordance with GAMP 5. Five evaluation levels were defined for detection and probability criteria. However, only three evaluation levels were recommended by the GAMP 5 guideline (see Appendix M3 of GAMP 5 [20]).

Regarding example 3. there was an intentionally judging risk as a low priority although the risk is high and other options available (e.g. software bug) (**DocTD.02**). In this use case, according to the sequential entry numbers, three identical malfunctions occurred at a safety device. The safety function was triggered due to operator intervention and software bugs. For each risk entry, function tests were defined, and the priorities were lowered from High/Median to Low. Below these three risk entries in the risk trace matrix, the fourth risk entry noted that a related malfunction occurred (e.g. safety function was not triggered), according to the sequence number. The impact documented in the fourth one was significant (e.g. operator safety might be impacted). However, in the first evaluation, the priority of the fourth entry was ranked at Low; thus, no second evaluation was performed, according to the internal risk assessment guideline. Therefore, a potential TD was identified from this use case.

3.2. Threats to Validity

First, there might be a bias when deriving the research questions and hypotheses. The research questions follow the ideas and suggestions from recent publications to reduce this bias.

Second, the engineering documents were provided by an industrial partner. There is a threat that if the participant did not offer authentic documents, the authors would not be able to figure this out. Nevertheless, an evaluation was conducted to confirm the results.

Third, during the analysis phase, the data (i.e. engineering documents) were analysed independently by the first author, then discussed with the second author to reduce data interpretation errors.

4. Overview of the analysis results

This section first reports an overview of the analysis results, followed by the typical TD use cases at aPS manufacturers. Following the idea of Li et al. [2], identified TD items from the use cases are classified into the phases in life cycles of aPS. Second, three use cases are selected and presented. Since the applied research method was a meta-analysis, the reported TD is mainly related to Documentation TD. Finally, an example of the delta between first and second risk evaluations in the GAMP 5 process is presented. The research questions, hypotheses, results and related sections are summarised in Table 1.

4.1. An enlargement of Technical Debt classification for validation of process control systems engineering

Based on the meta-analysis method introduced, an enlargement of TD classification in aPS manufacturing is achieved. There are 26 TD sub-types derived and generalised from 21 considered use cases. The TD sub-types are subsequently classified into 4 TD types. Twenty-five of 26 TD sub-types is new. The sub-types of TD are organised and presented in Fig. 2. The TD sub-types relating to the risk evaluation process are grouped into **Risk analysis TD** within **Design TD** and **Risk assessment TD** within **Documentation TD** (illustrated with thick border boxes in Fig. 2). It could be observed that most new TD sub-types belong to Design TD and Documentation TD since the work was a meta-analysis on engineering documents. As some identified TD relate to a specific engineering phase such as design or manufacturing, **RQ2.1** is addressed.

Table 1 Summary of the results addressing the three Research Questions

Research question	Sub research questions	Results	Related section
RQ1 How to analyse documents to study engineering TD?	**RQ1.1**: Engineering TD could be determined by analysing design specifications.	Approach 1 (bottom-up – start from specifications)	3
		Discrepancies between requirements and standards (**DocTD.04**)	4.2.2
	RQ1.2: Engineering TD could be identified by analysing the risk trace matrix document.	Approach 2 (top-down – start from risk analysis)	3
		Insufficient instructions for (sensor) adjustment in operation (**DocTD.13**)	4.2.3
RQ2 How are new detected TD systematically classified?	**RQ2.1**: Identified TD might relate to a specific engineering phase.	• New TD sub-types classified in Manufacturing TD • New TD sub-types classified in **Process design TD** or **Risk analysis TD** within Design TD	4.1
	RQ2.2: Identified TD may link to cross-engineering stages.	New TD sub-type **Risk assessment TD** within Documentation TD	4.1
		Insufficient instructions for (sensor) adjustment in operation (**DocTD.13**)	4.2.3
RQ3 To which extent of benefit the reported TD cases are evaluated by industrial experts?	**RQ3.1**: Reported TD cases could provide a soundness proof concerning engineering risks.	Reported TD cases are presented and confirmed by industrial experts.	4.4
	RQ3.2: Usefulness of reported TD cases could be perceived as a potential quality improvement.		

4.2 Examples of TD Use Cases

Due to confidentiality, the gained results are presented using two lab demonstrators. Firstly, the section introduces both demonstrators; the extended Pick and Place Unit (xPPU) [23], a community research demonstrator, and MyJoghurt [24] demonstrators. Secondly, the section describes three examples of TD use cases in detail.

4.2.1 Introduction of xPPU and MyJoghurt Demonstrators

The "extended Pick and Place Unit" (xPPU) represents a lab-size demonstrator in the SPP 1593 programme for production automation [23]. The xPPU consists of a stack providing the workpieces (WP), a crane to transport the WPs, a stamping unit, conveyors and ejectors to sort the different WP types into slides [23]. This paper focuses on the Linear Handling Module – Resorting Workpieces (cp. Fig. 3).

MyJoghurt [24] is a laboratory Cyber-Physical Production System. The MyJoghurt consists of multiple modules: (1) a conveyor module for bottle transportation, (2) a handling robot for handling and commissioning, (3) preparation or tank control process modules, and (4) two filling stations with storage modules and separators [24]. This paper focuses on the bottle separation feature of the conveyer module (cp. Fig. 4).

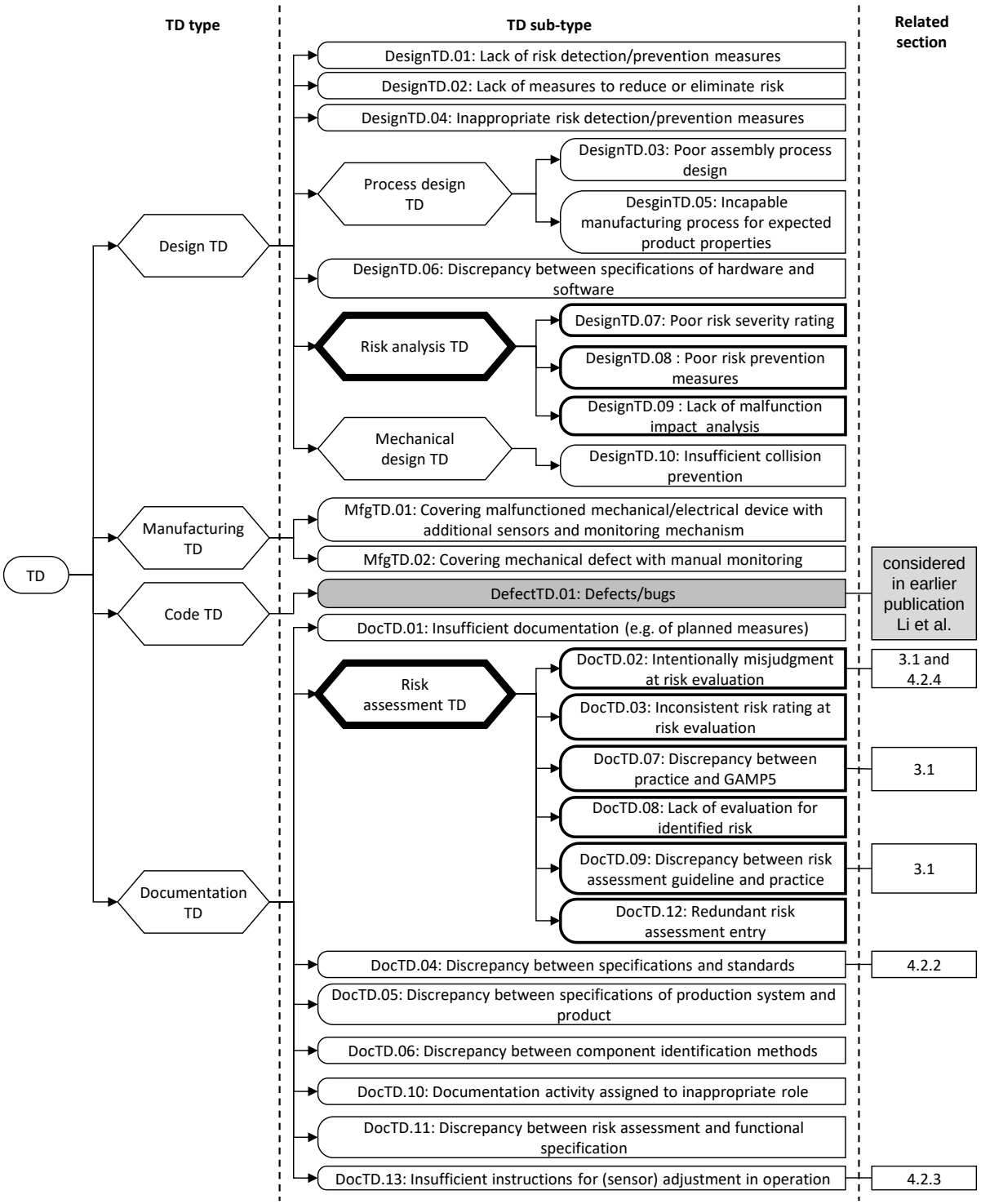


Fig. 2 Enlargement of Technical Debt classification for validation of process control systems' engineering

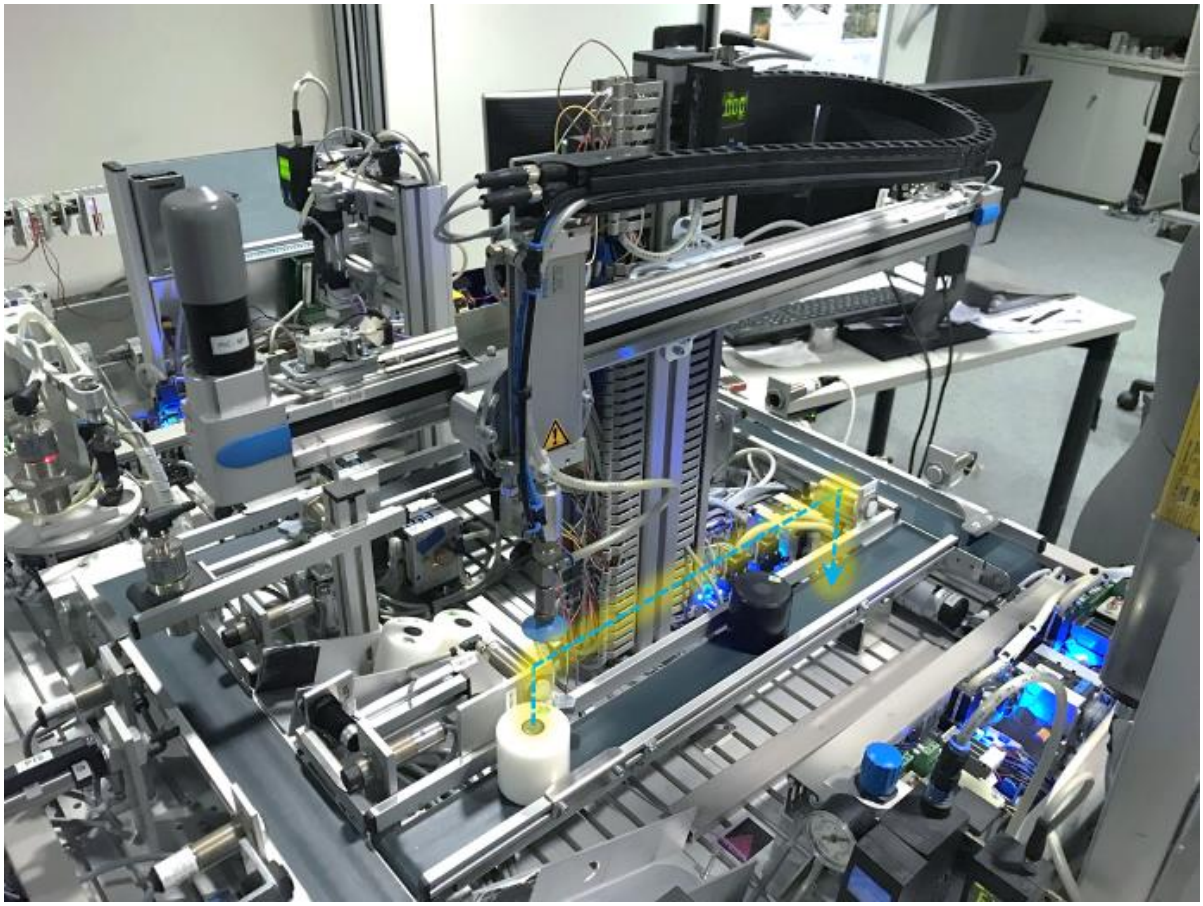


Fig. 3 xPPU demonstrator – Linear Handling Module – Resorting Workpieces

4.2.2 Example 1 – Specifications Versus Standards

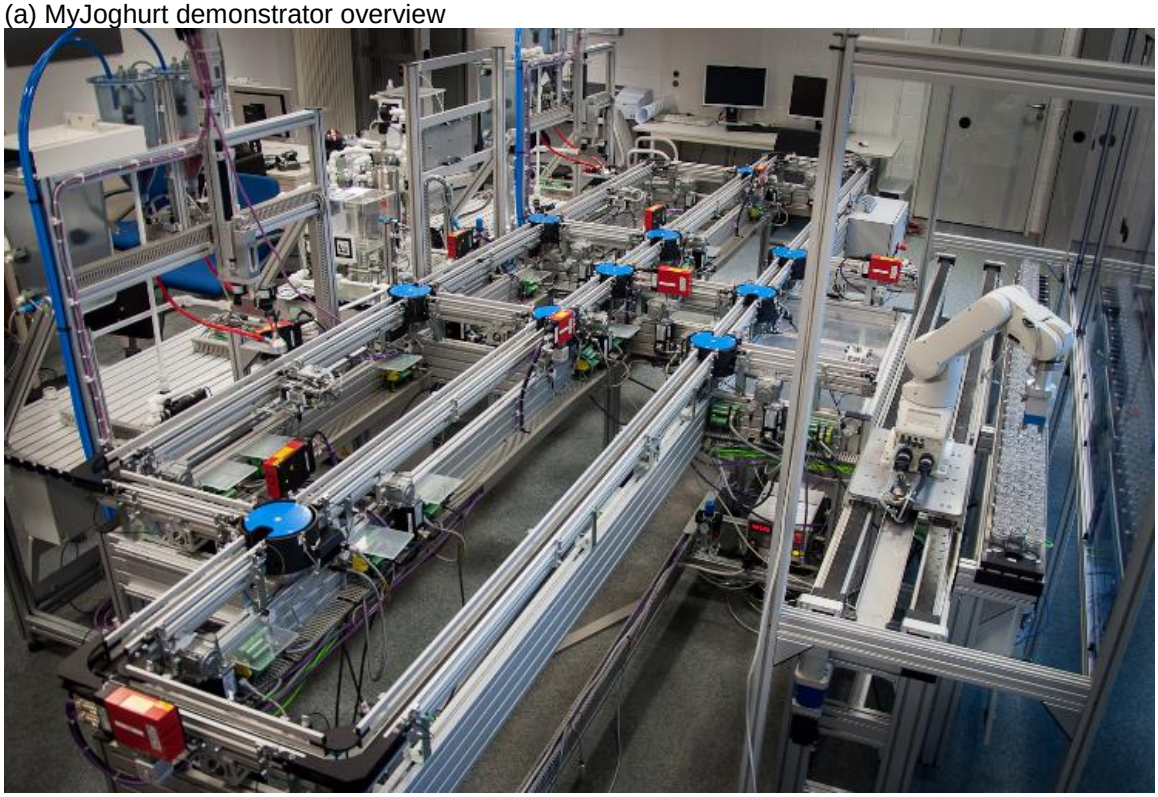
A comparison between design specifications and standards/industrial directives showed some discrepancies between requirements and standards (**DocTD.04**). For instance, the design specification used yellowish-green to represent the normal operating state (e.g. on HMI). However, according to VDI/VDE 3699, the yellowish-green is reserved for the prewarning state. In a follow-up discussion, industrial partners informed that they were aware of the reported issues, but the company could not resolve these discrepancies due to specific customers' wishes. In this case, the customer has forced the company to disobey standards. The partners also raised concerns that significant efforts may be required to correct or maintain the inconsistencies in different systems sold to other customers if deviations are large. As an examination of design specifications successfully identifies TD and confirmed by related experts, this example is proof of answer for **RQ1.1**.

4.2.3 Example 2 – Workpiece Separation Failure

The risk trace matrix document reported a sub-optimal manufacturing process due to insufficient sensor adjustments. A method is considered sub-optimal if certain desired qualities are not satisfied. For example, some NOK products could be assessed as OK ones due to insufficient instructions for (sensor) adjustment in operation (**DocTD.13**). Because of incorrect adjustments, the sensor delivers the wrong signal; thus, the NOK object is detected as OK (cp. Fig. 5a). Consequently, the NOK object is separated into the wrong lane (cp. Fig. 5b).

It is worth noting that the "incorrect adjustment" incidents appeared several times in the reported document. Due to "(sensor) adjustment" Documentation TD, some additional efforts might be required at other project phases such as assembly or operation. Therefore, the plant was operating at a sub-optimal level. As an analysis on the risk trace matrix document could support in identifying TD confirmed by related experts, **RQ1.2** is addressed.

(a) MyJoghurt demonstrator overview



(b) A bottle separation feature

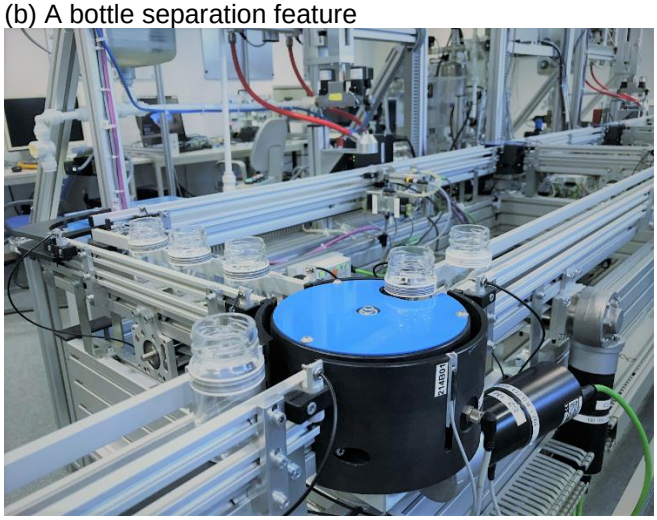


Fig. 4 MyJoghurt demonstrator – (a) overview, (b) bottle separation feature

This use case suggests further analysis should be performed regarding "(sensor) adjustment" to assist with risk assessment of change control in operation. Thus, the use case illustrates a typical example of risk management throughout the system life cycle (see section 5.3 of GAMP 5 [20]). As a cross-stage engineering TD is identified and confirmed, this example is proof of answer for **RQ2.2**. Also, there were similar use cases to this example, such as travel paths not being collision-free due to poor design practice.

4.2.4 Example 3 – Workpiece Manipulation Failure

This example relates to the workpiece manipulation process. A disordered workpiece needs to be manipulated (e.g. sorted back) (cp. Fig. 6). However, the handling module attempts inaccurately due to compressed air failure or workpiece density (cp. Fig. 6b1-2).

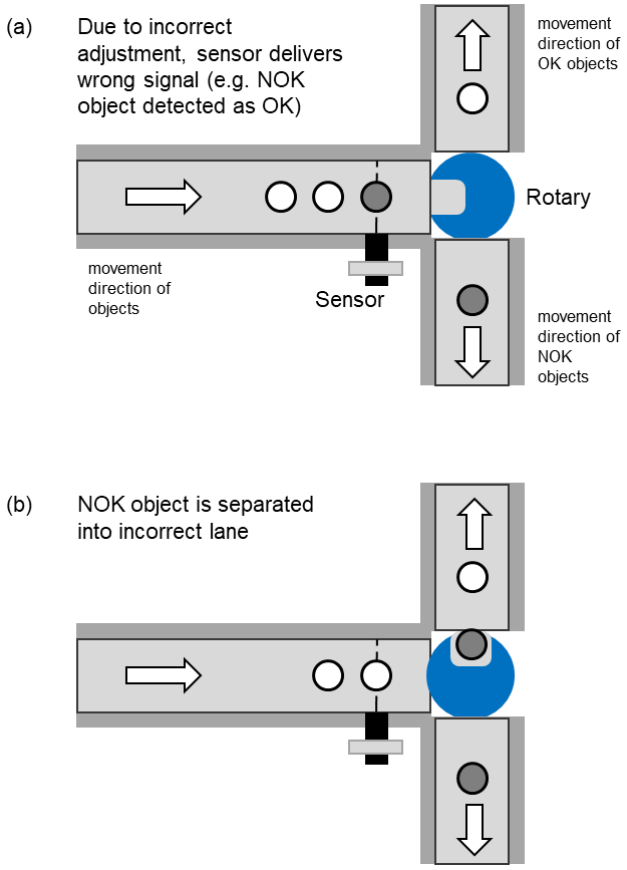


Fig. 5 Workpiece separation failure – (a) sensor delivering the wrong signal, (b) incorrect workpiece separation

The analysis identified an intentional misjudgment at the second evaluation (**DocTD.02**) in this use case. The risk trace matrix noted different detection/prevention strategies being applied at different levels, such as defect/failure in compressed air supply and bugs in the software (OK / NOK strategy). Thus, the root cause might not have been identified yet, but the risk priority was reduced from High to Low/Median, and some undesired additional procedures were added into SOP (Standard Operating Procedure) for the customer. For instance, when the pneumatic supply returns, the machine sends a message to check the production system and confirm the HMI. The operators would need to check the location of the parts before continuing production. Therefore, extra effort might be required (e.g. operator training) due to letting production compensate for the problem originating in the engineering phase.
There were two options to handle if this failure occurred. The first option could be an implementation of an automatic handler with a software update (e.g. temporarily changing conveyor direction to move back the grey workpiece and then performing the second attempt). The effort for software implementation was discussed in the previous work [19]. The second option, as presented, would be manual fault handling. There, the disorder workpiece is manually manipulated by the operator then needs to update the current state via HMI. The effort could be calculated using HTA (hierarchical task analysis), a technique widely used to analyse human tasks by constructing a hierarchical task list associated with a specific process [44]. Further work would be required to compare the efforts required for these two options.

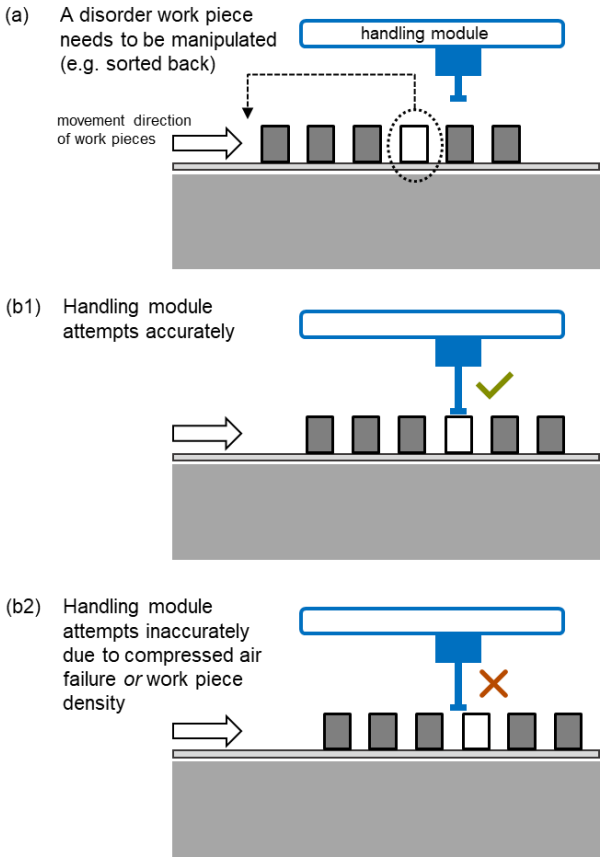


Fig. 6 Workpiece manipulation failure – (a) white workpiece need to be sorted back, (b1) successful attempt of handling module, (b2) failed attempt of handling module

4.3 Delta Between First and Second Risk Evaluations

On average, risk priorities are reduced from High to Low (cp. Fig. 7). "Incorrect adjustment" appears several times in risk analysis at different areas (purple columns). The delta between the evaluations (grey columns) seems to be pretty high (only risks having both evaluations considered). It should be noted that the system has three "Assembly" areas (A07, A13 and A16) and the most occurrences of "Incorrect adjustment" occur at those areas. The results indicate to which extent Documentation TD might have occurred at areas and where it appeared the most. Thus, further investigation is recommended in those areas.

4.4 An Evaluation with Industrial Experts

An evaluation was conducted with two industry experts from the company which provided the engineering documents to confirm the results. The presentation includes an introduction to the TD concept, delta between first and second risk evaluations (cp. Fig. 7), TD classification (cp. Fig. 2). Two selected examples (reported in sections 5.2.2 and 5.2.3) were presented and confirmed. Convinced by the present research results, the experts noted that they would need to conduct further discussions with quality personnel to review the significant difference between the two evaluation rounds. Thus, **RQ3.1** and **RQ3.2** were addressed.

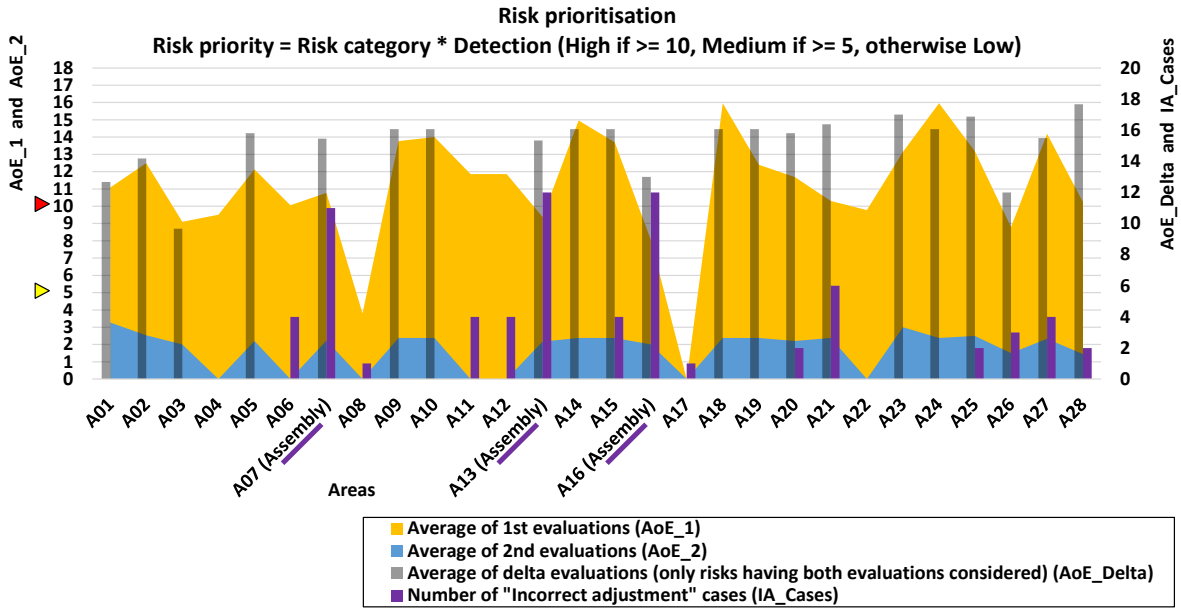


Fig. 7 High delta between two risk evaluation rounds and Documentation TD at assembly areas

5. Conclusion and Outlook

This study uncovers the contagious character of TD. TD may not only spread throughout the life cycle of a manufacturing system but may also burden the operating company and customer consuming yoghurt or a pharmaceutical product. The present TD items are associated with risks of the manufacturing systems in the healthcare sector, which requires specific safety-critical validation processes such as GAMP. Most of these systems often operate for several decades. Thus, the research results are critical as they could support avoiding harming lives and reducing costs for TD. The reported TD use cases were presented and confirmed by industrial experts. Therefore, the empirical evidence from this work contributes to body knowledge with the enlargement of TD classification with TD sub-types from validation of process control systems engineering that needs to undergo a validation process according to GAMP. The research outcome regarding risk evaluation shows areas to improve OEE for the studied manufacturing system from a TD perspective. TD might significantly impact the OEE for automated production systems; thus, machine and plant manufacturers should start monitoring TD more closely. TD identification step reported in this paper is a prerequisite for achieving further TD management goals, such as supporting decision-making to indicate whether TD items are acceptable or the decision needs to be changed.

The developed meta-analysis approaches can be reused as an analysis method for other work aiming at a similar purpose (i.e. TD identification using engineering documents). In addition, the presented meta-analysis techniques can be further applied to engineering documents in different domains. Thereby, the results from this work can be used as a reference as well as a benchmark.

Future work on TD in aPS can examine other documents such as manuals, maintenance logs or purchasing documents. The results could support aPS manufacturers and customers to increase OOE further; thus, improving manufacturing productivity.

6. References

[1] Cunningham W. The WyCash portfolio management system. ACM SIGPLAN OOPS Messenger. 1993;4(2):29-30. doi:10.1145/157710.157715.

[2] Li Z, Avgeriou P, Liang P. A systematic mapping study on technical debt and its management. J Syst Softw. 2015;101:193-220. doi:10.1016/j.jss.2014.12.027.

[3] Hegde V. Case study - Risk management for medical devices (based on ISO 14971). Proc - Annu Reliab Maintainab Symp. 2011. doi:10.1109/RAMS.2011.5754492.

[4] Avgeriou P, Kruchten P, Ozkaya I, Seaman C. Managing Technical Debt in Software Engineering (Dagstuhl Seminar 16162). *Dagstuhl Reports*. 2016;6(4):110–138. doi:10.4230/DagRep.6.4.110.

[5] Dong QH, Ocker F, Vogel-Heuser B. Technical Debt as indicator for weaknesses in engineering of automated production systems. *Prod Eng*. 2019;13(3-4):273-282. doi:10.1007/s11740-019-00897-0.
[6] Guo Y, Seaman C, Gomes R, et al. Tracking technical debt - An exploratory case study. In: *2011 27th IEEE International Conference on Software Maintenance (ICSM)*. ; 2011:528-531. doi:10.1109/ICSM.2011.6080824.
[7] Fairley RE, Willshire MJ. Better Now Than Later: Managing Technical Debt in Systems Development. *Computer (Long Beach Calif)*. 2017;50(5):80-87. doi:10.1109/MC.2017.124.
[8] Ramasubbu N, Kemerer CF. Managing technical debt in enterprise software packages. *IEEE Trans Softw Eng*. 2014;40(8):758-772. doi:10.1109/TSE.2014.2327027.
[9] Holvitie J, Leppanen V. Examining technical debt accumulation in software implementations. *Int J Softw Eng its Appl*. 2015;9(6):109-124. doi:10.14257/ijseia.2015.9.6.12.
[10] Kruchten P, Nord RL, Ozkaya I. Technical Debt: From Metaphor to Theory and Practice. *IEEE Softw*. 2012;29(6):18-21. doi:10.1109/MS.2012.167.
[11] Ampatzoglou A, Ampatzoglou A, Chatzigeorgiou A, Avgeriou P. The financial aspect of managing technical debt: A systematic literature review. *Inf Softw Technol*. 2015;64:52-73. doi:10.1016/j.infsof.2015.04.001.
[12] Snipes W, Robinson B, Guo Y, Seaman C. Defining the decision factors for managing defects: A technical debt perspective. In: *2012 Workshop on Managing Technical Debt (MTD)*. IEEE; 2012:54-60. doi:10.1109/MTD.2012.6226001.
[13] Ozkaya I, Kruchten P, Nord R, Brown N. Second international workshop on managing technical debt. In: *Proceeding of the 33rd International Conference on Software Engineering - ICSE '11*. ACM Press; 2011:1212. doi:10.1145/1985793.1986051.
[14] Vogel-Heuser B, Rösch S, Martini A, Tichy M. Technical debt in Automated Production Systems. In: *2015 IEEE 7th Workshop on Managing Technical Debt*. IEEE; 2015:49-52. doi:10.1109/MTD.2015.7332624.
[15] A. Martini and J. Bosch, "The Danger of Architectural Technical Debt: Contagious Debt and Vicious Circles," in *2015 12th Working IEEE/IFIP Conference on Software Architecture*, 2015, pp. 1–10.
[16] Martini A, Bosch J. The Danger of Architectural Technical Debt: Contagious Debt and Vicious Circles. In: *2015 12th Working IEEE/IFIP Conference on Software Architecture*. IEEE; 2015:1-10. doi:10.1109/WICSA.2015.31.
[17] Vogel-Heuser B, Bi F. Interdisciplinary effects of technical debt in companies with mechatronic products — a qualitative study. *J Syst Softw*. 2021;171:110809. doi:10.1016/j.jss.2020.110809.
[18] Besker T, Martini A, Bosch J, Tichy M. An investigation of technical debt in automatic production systems. In: *Proceedings of the XP2017 Scientific Workshops on - XP '17*. ACM Press; 2017:1-7. doi:10.1145/3120459.3120466.
[19] Vogel-Heuser B, Rösch S, Fischer J, Simon T, Ulewicz S, Folmer J. Fault Handling in PLC-Based Industry 4.0 Automated Production Systems as a Basis for Restart and Self-Configuration and Its Evaluation. *J Softw Eng Appl*. 2016;09(01):1-43. doi:10.4236/jsea.2016.91001.
[20] ISPE. : A Risk-based Approach to Compliant Gxp Computerized Systems. 2008. ISBN 1931879613.
[21] Boehm B. Verifying and Validating Software Requirements and Design Specifications. *IEEE Softw*. 1984;1(1):75-88. doi:10.1109/ms.1984.233702.
[22] Lueddemann T. A relational database for computer assisted documentation of medical products. *PhD thesis*. 2022. Online: https://mediatum.ub.tum.de/1553512. Accessed March 10, 2022.
[23] Pick and Place Unit. Online: https://www.mec.ed.tum.de/ais/forschung/demonstratoren/ppu/. Accessed March 10, 2022.
[24] MyJoghurt – involved in the Platform Industrie 4.0's Germany-wide online roadmap "Industrie 4.0". Online: http://i40d.ais.mw.tum.de/. Accessed March 10, 2022.
[25] NAMUR – Interessengemeinschaft Automatisierungstechnik der Prozessindustrie e.V. Namur.net. Online: https://www.namur.net/. Accessed March 10, 2022.
[26] PackML. Omac.org. Online: https://www.omac.org/packml. Accessed March 10, 2022.
[27] Neuhaus H, Kremers H, Karrer T, Traut R. Validation of the process control system of an automated large scale manufacturing plant. *Pharm Acta Helv*. 1998;72(6):333-342. doi:10.1016/s0031-6865(97)00029-0.

[28] Früh KF, Schaudel D, Urbas L, Tauchnitz T. Handbuch der Prozessautomatisierung. 2014. ISBN 3835633724.
[29] Bougouffa S, Dong QH, Diehm S, Gemein F, Vogel-Heuser B. Technical Debt indication in PLC Code for automated Production Systems: Introducing a Domain Specific Static Code Analysis Tool. *IFAC-PapersOnLine*. 2018;51(10):70-75. doi:10.1016/j.ifacol.2018.06.239.
[30] Glennon B. Control system validation in multipurpose biopharmaceutical facilities. *J Biotechnol*. 1997;59(1-2):53-61. doi:10.1016/S0168-1656(97)00164-8.
[31] Jatto E, Okhamafe A. An Overview of Pharmaceutical Validation and Process Controls in Drug Development. *Trop J Pharm Res*. 2002;1(2):115. doi:10.4314/tjpr.v1i2.14592.
[32] Aleem H, Zhao Y, Lord S, McCarthy T, Sharratt P. Pharmaceutical process validation: An overview. *Proc Inst Mech Eng Part E J Process Mech Eng*. 2003;217(2):141-151. doi:10.1243/095440803766612801.
[33] Smith ED, Szidarovszky F, Karnavas WJ, Bahill AT. Sensitivity Analysis, a Powerful System Validation Technique. *Open Cybern Syst J*. 2008;2(1):39-56. doi:10.2174/1874110x00802010039.
[34] Feigenbaum A, Scholler D, Bouquant J, et al. Safety and quality of food contact materials. Part 1: Evaluation of analytical strategies to introduce migration testing into good manufacturing practice. *Food Addit Contam*. 2002;19(2):184-201. doi:10.1080/02652030110053002.
[35] Alam S. Pharmaceutical process validation: An overview. *J Adv Pharm Educ Res*. 2012;2(4):185-200.
[36] Chowdary B V., George D. Improvement of manufacturing operations at a pharmaceutical company. *J Manuf Technol Manag*. 2011;23(1):56-75. doi:10.1108/17410381211196285
[37] Harpreet K, Gurpreet S, Nimrata S. Pharmaceutical Process Validation : A Review. *J Drug Deliv Ther*. 2013;3(4):189-194.
[38] Suseno BD. The strength of justified knowledge sharing on good manufacturing practices: Empirical evidence on food beverage joint venture company of Japan – Indonesia. *Qual - Access to Success*. 2019;20(170):130-135.
[39] Samson Y. Computerized System Validation: Regulatory Compliance and Process Safety in the Pharmaceutical Industry. *3rdEuropean Congr ERTS - Embed Real Time Softw*. 2006.
[40] Chen X, Voigt T. Implementation of the Manufacturing Execution System in the food and beverage industry. *J Food Eng*. 2020;278(August 2019):109932. doi:10.1016/j.jfoodeng.2020.109932.
[41] Meyliana, Fernando E, Surjandy, Marjuky. The Business Process of Good Manufacturing Practice Based on Blockchain Technology in the Pharmaceutical Industry. In: *2021 Fifth International Conference on Information Retrieval and Knowledge Management (CAMP)*. IEEE; 2021:91-95. doi:10.1109/CAMP51653.2021.9498104.
[42] Andriani DP, Aini APN, Lestari M, Purba P. Good manufacturing practices for risk management in food safety sustainability: An empirical study. *IOP Conf Ser Earth Environ Sci*. 2021;733(1). doi:10.1088/1755-1315/733/1/012118.
[43] Huysentruyt K, Kjoersvik O, Dobracki P, et al. Validating Intelligent Automation Systems in Pharmacovigilance: Insights from Good Manufacturing Practices. *Drug Saf*. 2021;44(3):261-272. doi:10.1007/s40264-020-01030-2.
[44] Annett J. Hierarchical task analysis. Handbook of cognitive task design. 2003; 2:17-35.

Paper6

at – Automatisierungstechnik.
https://doi.org/10.1515/auto-2022-0146
(*accepted on 04 May 2023*)

Applications

Quang Huan Dong*, Birgit Vogel-Heuser and Eva-Maria Neumann

Semi-automatic assessment of lack of control code documentation in automated production systems

Semiautomatische Bewertung fehlender Dokumentation von Steuerungscode in automatisierten Produktionssystemen

A risk-based approach to indicate documentation debt

Ein risiko-basierter Ansatz zur Indikation von Dokumentationsschulden

https://doi.org/10.1515/auto-2022-0146
Received November 11, 2022; accepted May 4, 2023

Abstract: This paper first examines the current state of industrial practice of documentation in automated production systems based on a large-scale survey in machine and plant manufacturing proving that companies still face major challenges in documentation. However, insufficient documentation creates friction since it may increase the risk of malfunction and high costs, and impede system development due to lack of traceability, especially for control software being one of the main functionality carriers. Therefore, secondly, a risk priority indicator RPI4DD is proposed to systematically capture the lack of control code documentation to avoid undesired costs due to inadequate documentation.

Keywords: automated production systems; documentation; machine and plant manufacturing; risk analysis; risk priority indicator; software engineering.

***Corresponding author: Quang Huan Dong**, Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany; and Vietnamese-German University, Thu Dau Mot City, Vietnam, E-mail: huan.dong@tum.de
Birgit Vogel-Heuser, Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany; Core Member of MDSI and Lead of Sector Work of MIRMI, Technical University of Munich, Munich, Germany; and Munich Institute of Integrated Materials, Energy and Process Engineering, Technical University of Munich, Garching, Germany, E-mail: vogel-heuser@tum.de
Eva-Maria Neumann, Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany, E-mail: eva-maria.neumann@tum.de

Zusammenfassung: Anhand einer groß angelegten Studie im Maschinen- und Anlagenbau werden der aktuelle Stand der industriellen Praxis bei der Dokumentation in automatisierten Produktionssystemen erläutert und die resultierenden Herausforderungen herausgearbeitet. Unzureichende Dokumentation erzeugt Reibungseffekte, da das Risiko von Fehlfunktionen und hohen Kosten gesteigert und die Systementwicklung aufgrund mangelnder Nachvollziehbarkeit behindert wird, insbesondere in der Steuerungssoftware als einem der Hauptfunktionsträger. Daher wird ein Risikoprioritätsindikator RPI4DD vorgeschlagen, um die Schwächen der Dokumentation in der Steuerungssoftware systematisch zu erfassen.

Schlagwörter: automatisierte Produktionssysteme; Dokumentation; Maschinen- und Anlagenbau; Risikoanalyse; Risikoprioritätszahl; Softwareentwicklung.

1 Introduction and motivation

In factory automation, machine (MM) and plant manufacturing (PM) companies are facing numerous challenges, such as high variability, Industry 4.0 requirements (e.g., small lot sizes) or long lifecycles up to decades [1, 2]. German manufacturers in this area, who used to be world-leading exporters, are confronted with growing competition worldwide and thus are struggling to stay competitive, particularly regarding work costs. Some MM and PM companies have to consider relocating engineering divisions to lower-wage countries with less experienced/qualified engineers who need more mature documentation [3]. Moreover, the scope of system functionalities realised by software is increasing, leading to a growing software complexity [4].

Thus, companies need to improve complexity management methods for software.

Fischer et al. [5] indicate that software complexity strongly influences comprehensibility (readability), a prerequisite to maintaining or reusing existing code. Thus, necessary documentation artefacts (e.g., architecture documents or code comments) are required to ease the readability of complex software. The impact of insufficient documentation might be significant, especially among low-skilled engineers in lower-wage countries where the engineering departments are relocated [3]. A motivational example is introduced in the following:

> In the commissioning phase […] an error handling routine for a pneumatic cylinder is missing […] In the optimal case, a change request to the programmer, who developed the library function for controlling pneumatic cylinders, should be made. However, as the time pressure to start the [systems] is high, the commissioner implements the error handling on the next higher software architecture level that he may access, i.e., directly in the application, since the library function is not accessible for him […] This conscious decision of avoiding proper change management and violating the architectural concept is often accompanied by a lack of documentation. As the commissioner works as fast as possible to start up the [systems], she/he does not document the changes made in the software resulting in many software versions in the field [6].

Motivated by the example above, a question arises: How do different factors influence control code documentation? Since the development characteristics of MM and PM are typically distinct, they often follow different engineering practices. Previous studies indicated that software analysis alone is insufficient to measure software quality (e.g., maturity level [7]) or support software evolution since automation software is part of a mechatronics system involving multiple disciplines or stakeholders [8]. Instead, the development process characteristics must be considered. Therefore, a web questionnaire is developed to study software maturity, complexity, and documentation in MM and PM [7].

The main contribution of this work is to address selected aspects of (semi-) automatically identifying a lack of control code documentation. First, based on the results of [7], a study is conducted to analyse the documentation aspect from expert responses to the questionnaire, especially regarding engineering practices or the automatic generation of information in MM and PM. Second, a concept to assess risk associated with insufficient documentation is proposed, and the concept's applicability to control code documentation is presented for supporting an early reaction or counter-measures in the development workflow.

The remainder of the paper is structured as follows. Section 2 provides the background and related work, followed by an analysis of industrial practice regarding documentation in MM and PM in Section 3. The concept and applicability of a risk-based approach to indicate insufficient documentation are presented in Section 4. Finally, Section 5 concludes the paper and provides an outlook on future work.

2 Background and related work

This section introduces some backgrounds of the systems developed by MM and PM so-called automated production systems (aPS), according to [8], followed by a description of the quality analysis of aPS software. The section continues with a discussion on the control code documentation. Concluding, the research gap is presented.

2.1 Development of aPS

aPS is developed by engineers from multiple disciplines [8]. The development often starts in the mechanical engineering discipline, which creates the construction plan of the mechanical parts. Based on this construction plan and component lists (e.g., sensors or actuators), the electrical engineers design the electrical system. Documents from both disciplines are then forwarded to the software engineers, who use them to develop the software running in Programmable Logic Controllers (PLC) (i.e. control software or aPS software), the hardware platforms used to manage the automation for machines and plants via sensor inputs and actuator outputs.

There are differences in the languages for PLCs compared to those used in classical software engineering. The aPS software is often developed following the IEC 61131-3 standard [5]. IEC 61131-3 defines three types of Program Organization Units (POU): (1) Function (FC), (2) Function Block (FB), and (3) Program (PRG). Each POU includes a comment header, a variable declaration section, and an implementation section. IEC 61131-3 compliant languages include three graphical languages (i.e., Ladder Diagram LD, Function Block Diagram FBD, and Sequential Function Chart SFC) and two textual languages (i.e., Structured Text ST and Instruction List IL). In the field of PLC programming, there are guidelines from the communities of PLCopen or MISRA [9]. PLCopen is a worldwide initiative of different platform suppliers to reduce engineering effort and enhance software quality by providing standards, guidelines, and education for platform users in the community of industrial automation. The PLCopen guidelines, e.g., naming conventions or structuring with SFC, support in ensuring, e.g., consistency of automation software. MISRA provides rules

and directives for coding and documentation that improve software maintainability. The MISRA guidelines, e.g., using start and end comment markers, support in, e.g., portability of automation software.

The aPS development is often validated with GAMP (Good Automated Manufacturing Practice) [10], which is a risk-based approach widely used in industry to regulate computerised systems. Risk Priority in the GAMP method (i.e., criticality estimation in risk analysis) is based on three factors *Severity*, *Probability*, and *Detectability* (cf. Figure 1). For each risk entry, ratings are given to each of the factors. Risk Priority is calculated with a multiplicative approach: factors are multiplied by each other in two stages. The factors are allocated to two levels: *Detectability* and *Risk class* are level 1 factors; *Severity* and *Probability* are level 2 factors. In the first stage, the *Severity* is offset against the *Probability*. This calculation determines the *Risk class* (1–3). In the second stage, the *Risk class* is then offset against the *Detectability* and results in a *Risk Priority* (low, medium, high).

2.2 Quality analysis of aPS software

In classical software engineering, various approaches are available to measure software characteristics, such as complexity or code documentation (i.e., code comments). It is due to high complexity or poor code documentation that might hinder software maintainability, which may introduce undesired additional costs [11]. Analogously, characteristics of aPS software need to be analysed to measure the criticality of documentation for the software. Vogel-Heuser et al. [12] proposed the MICOSE4aPS approach to measure the maturity of software considering different change types (e.g., bug fixing or new development). Wilch et al. [13] presented a semi-automatic concept to identify POU functionality based on characteristics of implementation or description. Fischer et al. [5] proposed an approach to compare the complexity of POUs using a metric for the *Overall Complexity* based on six sub-metrics, M_1–$M_{6,}$ as follows.

- ProgramLength (M_1) refers to Halstead's Program Length, i.e., the sum of the tokens (operators and operands).
- Cyclomatic Complexity (M_2) refers to McCabe's Cyclomatic Complexity determined by the number of decision statements.

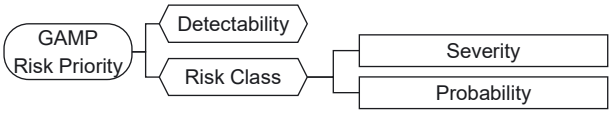


Figure 1: Factor hierarchy in the GAMP method [10].

- FanIn_FanOut (M_3) is determined by incoming (e.g., input variables) and outgoing (e.g., output variables) data flows.
- Vocabulary Size (M_4) and Difficulty (M_5) follow the ideas of Halstead's Vocabulary Size and Difficulty, which are also based on the number of (unique) operators and operands.
- Data Structure (M_6) targets the complexity of the data processed. For instance, the program interface variables add more complexity than local variables and sub-variables.

The calculation includes three steps. First, six metrics M_1–M_6 (M_i in (1)) are applied to each POU, resulting in six values with different scales per unit. The median values $\tilde{M}_1$ to $\tilde{M}_6$ are then determined. Second, the metrics values are scaled using equation (1) to the corresponding median.

$$C_{\mathrm{rel},M_i}(\mathrm{POU}) = \frac{M_i(\mathrm{POU})}{\tilde{M}_i} \cdot 100\% \tag{1}$$

Third, the *Overall Complexity* OC_rel of a POU is calculated based on the sum of the six metrics values scaled in the second step (cf. equation (2)). The weights w_i are used to adjust the influences of individual metrics on the overall complexity. The sum of w_i is 1 to scale Overall Complexity values to a predefined value range (e.g., (0, 1]).

$$OC_\mathrm{rel}(\mathrm{POU}) = \sum_{i=1}^{6} w_i \cdot C_{\mathrm{rel},M_i}(\mathrm{POU}) \tag{2}$$

2.3 Lack of control code documentation in the aPS domain

In classical software engineering, the phenomenon of insufficient documentation is referred to as technical debt (TD), more specifically, documentation debt [14]. The TD concept describes a context where a technical compromise is selected against better knowledge [15]. For instance, low-quality code is delivered to meet an urgent deadline. The technical compromise can enable a short-term benefit (e.g., quicker delivery) but may introduce some undesired effects (e.g., high maintenance cost). The survey of Li et al. [15] indicates that although documentation debt received significant attention, studies on documentation debt management are scarce. Avgeriou et al. [16] reported that TD always relates to cost. Inadequate monitoring of TD may introduce additional long-term effort in software projects. TD might cause project delivery delays, as well as a decrease in organisational profitability. Detofeno et al. [17] suggested that to prioritise TD in software, one should not only consider source code alone but also consider external factors such as project characteristics or test coverage.

Near the aPS domain, Martini et al. [18] analysed the accumulation of TD in software in embedded systems and additional activities (e.g., refactoring) required at five software companies. They reported that insufficient documentation "causes the misinterpretation by the developers implementing code" as well as hinders refactoring activities. However, code comments were not considered as the work focused on the level of design and architecture. A survey with six software companies from Besker et al. [11] indicated that time pressure commonly triggers documentation debt, such as insufficient updating related documentation for code implementation. The study reported a significant productivity loss of software developers due to TD. However, the code comment was not focused.

In the aPS domain, Besker et al. [14] investigated the work of software engineers at one company working with aPS in Scandinavia and reported that there were "quite a lot of resources in paying the interest on Technical Debt, on average 32 % of the development time". Biffl et al. [19] studied the risks of TD in engineering artefacts related to data exchange (e.g., process documentation and configuration management). Waltersdorfer et al. [20] reported two types of documentation debt: Insufficient Data Model and Insufficient Product Process Resources Model. However, the code comment was not considered in the above studies.

Due to the nature of aPS, on-site changes are often performed to adapt the manufacturing systems to unanticipated raw materials or environmental conditions. Under pressure to start up the system, proper control code documentation for on-site adaptions might be neglected, resulting in documentation debt (cf. Section 1, e.g.). TD is causing additional development and maintenance costs over time [14] or developer productivity loss [11]. Due to the long lifetime of aPS, the impact from TD might be large. Unfortunately, awareness of TD in the aPS domain is still low in general, according to a survey of 48 German companies supplying aPS [21].

2.4 Research gap

Research addressing aPS documentation quality is still scarce. Among the publications, Neumann et al. [22] introduced templates to document design decisions at the architectural level of aPS. Neumann et al. [23] proposed some simple metrics such as HeaderCommentLinesOfCode (size of comment header), MultiLineComments (amount of comments wrapping over multiple lines), and SourceCodeCommentedRatio (density of comments) to assess the software maintainability. The authors' previous work [7] mainly focused on methods for document exchange or on how code/configuration is generated from documents.

As aforementioned, to assist the engineers in maintaining, enhancing, or reusing the control code at a later stage, the comment header often provides an overview description of the POU, and the comments in the variable declaration and implementation section explain the control code in detail. A well-documented software supports its reusability in future projects; thus, the development costs could be optimised. Not only code comments, but a broad view should also be considered due to interdisciplinary constraints. Thus, the availability of documentation in different disciplines, the use of internal guidelines, or the automatic generation of engineering documents needs to be studied. Another factor in enabling reusability is standardisation, i.e., whether standardised guidelines provided by the community are followed. To the best of the authors' knowledge, there is no method to determine the criticality or quality of documentation in the aPS domain. By analysing results from an industrial expert survey in machine and plant manufacturing, this work firstly studies the state of the practice regarding documentation. Secondly, factors influencing the documentation activities are derived considering the documentation debt concept to propose a risk priority indicator for identifying the documentation debt. An overview of findings from related work and expert survey corresponding to the sections is illustrated in Figure 2.

3 An analysis of state of the practice regarding documentation

The current state of practice in industry regarding documentation was surveyed using a questionnaire distributed via newsletters and web pages addressing a German-speaking community interested in embedded systems, mechanical, electrical or software engineering in MM and PM. For this paper, the focus is on the documentation aspect of the questionnaire, which was not the target of the previous investigation [7]. Hereafter, #[number] denotes a question with the ordinate number in the questionnaire available online [24]. The eight questions used in this analysis are listed in the Appendix. Among 322 companies that started to answer the questions, 146 finished it. As this work targets aPS, those 146 completed cases are firstly categorised by their answers regarding industrial sectors (#65). Thus, only responses from companies assigning themselves to at least one of the sectors associated with MM and PM, such as material handling or woodworking machinery, are selected. This reduces the cases from 146 to 71. Second, only companies assigned themselves to MM or PM in #69 are considered,

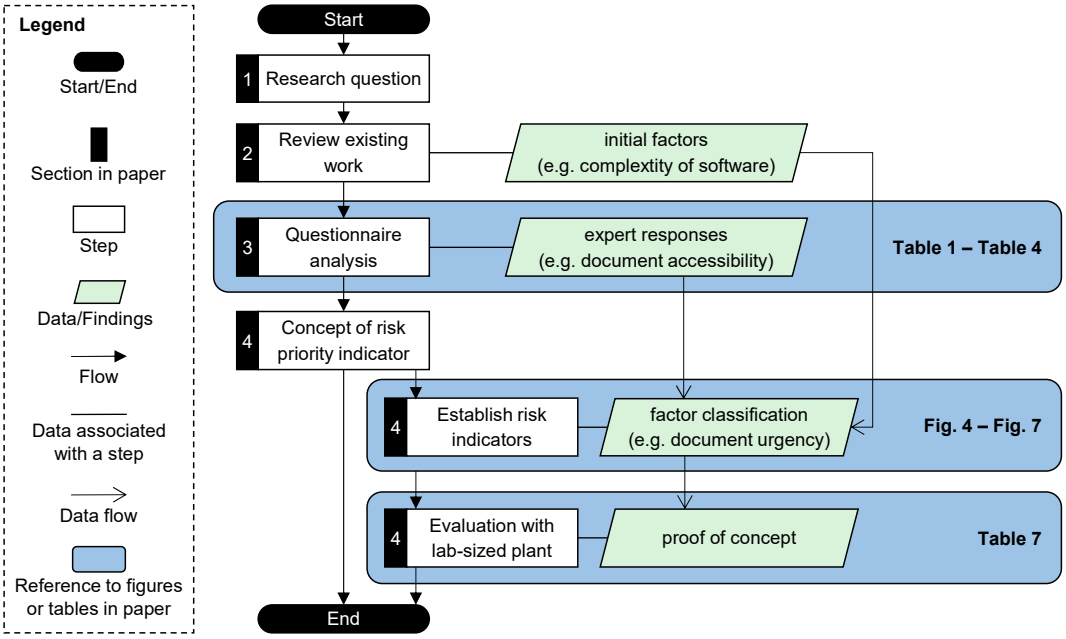


Figure 2: Steps of the study and findings correspond to the sections.

thus reducing the cases from 71 to 61 companies, representing the study's base data. In the following, the results of the documentation aspect are reported.

As a pre-processing step, question #66 surveyed the company size. Question #72 was used to study the functional description availability in disciplines. The availability of documents is lowest in software compared to electrical/electronics and mechanical disciplines (cf. Table 1). Across the disciplines, the availability is highest at large companies (>1000 employees) and lowest at small-and-medium companies (from 50 to 250 employees). The availability of documents "when design begins" or "provided at milestones" is lowest in software (cf. Table 1). Thus, document availability is mostly poor when engineering begins and varies later depending on the discipline and company size.

Questions #51 and #52 are used to study how engineering artefacts are automatically generated in different disciplines (e.g., code generation or configuration). The identified sources reveal that requirements documents are most commonly used across all disciplines (cf. Table 2); however, the usage frequencies are still low. Thus, automatically generating engineering artefacts is still poor in general.

Following guidelines or checklists in software engineering supports high-quality code and traceable software [9]. Question #77 studied the differences regarding the levels of detail of internal guidelines in different disciplines. Software and electrics/electronics received more precise instructions (21 % and 20 %, respectively) than mechanics (11 %) (cf. Table 3). However, the formulation of guidelines is still poor in general.

Question #34 examined the usage of PLCopen and company-specific guidelines. The responses of #34 show a low frequency of usage of PLCopen guidelines (10 %) compared to in-house guidelines (cf. Table 4). The internal

Table 1: Evaluation of expert surveys on document accessibility across disciplines (#72 [24]).

Availability of documents	Discipline		
	Software	Electrics/electronics	Mechanics
When design begins	9 %	13 %	13 %
Often late	28 %	29 %	25 %
Provided at milestones	25 %	31 %	25 %
Engineer needs to request	20 %	8 %	13 %
Engineer has to collect details from customer	3 %	4 %	3 %
Not applicable	15 %	15 %	20 %

Table 2: Evaluation of expert surveys on automatically generating engineering artefacts across disciplines (#51 and #52 [24]).

Source for automatic generation	Discipline		
	Software	**Electrics/electronics**	**Mechanics**
Requirements documents	38 %	29 %	26 %
Component lists	–	11 %	11 %
E-Plan[a]	–	17 %	7 %
Models	12 %	–	–
Total from above artefacts	***50 %***	***57 %***	***44 %***
Nothing	16 %	17 %	18 %

[a]E-Plan: a tool for electrical engineering of machines and plant systems.

Table 3: Evaluation of expert surveys on detail levels of in-house guidelines across disciplines (#77 [24]).

Detail level of internal guidelines	Discipline		
	Software	**Electrical/electronics**	**Mechanics**
Poor in-house guidelines	33 %	36 %	44 %
Fine-grained in-house guidelines	31 %	29 %	24 %
Precise instructions	21 %	20 %	11 %
Not applicable	15 %	15 %	21 %

Table 4: Evaluation of expert surveys on usage of PLCopen and company-specific guidelines in software discipline (#34 [24]).

Types of guidelines in use	Response
PLCopen guidelines	10 %
In-house guidelines	80 %
None	10 %

guidelines are employed by a large portion of the surveyed companies (80 %); thus, the usage of guidelines from the community is less than company-specific guidelines. Furthermore, the 10 % remaining responses reported that no guideline is being followed.

In summary, the results show a ***lack of availability of required documents, low-moderate automatic generation of information in engineering, a lack of exact instructions, and high reliance on in-house guidelines*** in MM and PM. These findings reveal the potential triggers of documentation debt in aPS. Thus, a methodology to indicate the risk of documentation debt is necessary.

4 Concept of indicators for control code documentation debt

The expert responses presented in Section 3 emphasise that insufficient documentation is still a critical challenge in industrial practice of MM and PM companies. The quantitative results in Section 3 are substantiated by a qualitative study [25] conducted by the authors with an industrial partner developing aPS for the healthcare industry. The results indicate a high need to improve document availability as an essential to build up a cross-disciplinary development process. Especially in software, which must be adaptable over many years and is increasingly becoming one of the main carriers of system functionality, a lack of documentation is a significant cost factor. However, to systematically improve the software documentation, it is first necessary to quantify the risk of insufficient documentation and thus to assess where an improvement in documentation is most urgent to prevent errors at an early stage, reduce engineering duration, and shorten the time-to-market. Therefore, a Risk Prioritisation Indicator of Documentation Debt (i.e. RPI4DD) is introduced in the following to quantify the risk of a lack of control code documentation.

This section first derives requirements of indicators for documentation debt, then presents an approach to transfer the concept of RPI4DD, followed by a set of selected factors to indicate control code documentation debt in aPS. The selected factors are based on current literature and expert responses identified in Section 3. The section continues with a description of the proof of concept of RPI4DD, including an evaluation using a lab-sized plant. Finally, a summary of RPI4DD calculation and requirement fulfilment is given.

4.1 Derived requirements of indicators for documentation debt

From the state of the art, requirements of indicators for documentation debt are derived in the following.

R1. RPI4DD should be extensible with new factors identified. The requirement for extensibility is due to TD research in the aPS domain being an ongoing work. The aspects of identifying TD, in particular documentation debt, are not fully explored yet. Thus, the list of factors is not yet fixed. When new factors are identified in future work, their calculation rules should be seamlessly integrated into RPI4DD.

R2. RPI4DD should be flexible with calculation rules of individual factors. Different calculation rules might be introduced to assess a factor due to the high diversity of applications in the context of the aPS domain. Therefore, flexibility is a requirement for RPI4DD.

R3. RPI4DD should be automatable. The program size of industrial projects is often large; thus, an automatic method is required to enable applicability in industrial practice.

4.2 Concept of a risk-based approach to indicate documentation debt

The concept of determining indicators for documentation debt is based on the GAMP method, which allows extensibility for additional factors that could be added to an appropriate level. Therefore, since the GAMP approach satisfies ***R1 (extensible)***, the approach is used as a basis to derive RPI4DD. To satisfy ***R2 (flexible)***, weights could be added to reflect the different importance of each factor. In addition, besides the multiplication operator, other operators (e.g., addition) could also be applied to the calculation of the factors.

In the following, based on the GAMP method, the six main steps to establish a new risk assessment method for documentation debt (cf. Figure 3) are described. This work focuses as a starting point on the control code documentation, in particular code comments and naming conventions. However, the design of RPI4DD shall enable the integration of further documentation artefacts, such as manuals or architect documents, in future work (cf. R1).

- *Step 1: Identify an initial set of factors influencing TD.* In documentation in general and control code documentation in particular, a successful document often meets two criteria: (1) the document covers the necessary information of the object being described (i.e. *Document Coherence*) and (2) the document is available on time (i.e., *Document Urgency*).
- *Step 2: For each identified factor, identify the sub-factors influencing the factor.* GAMP method stops at level 2 factors (cf. Figure 1); however, this step can be further applied for sub-factors if necessary. It should be noted that the ratings in the GAMP method may cause confusion. For example, a *Risk class* rating of 3 and low *Detectability* result in low *Risk Priority*, while a *Risk class* rating as 1 and low *Detectability* result in high *Risk Priority.* This paper does not aim to address the above issue but just proposes some modifications regarding the ratings. Still, the offset mechanism of the GAMP method is followed by grouping related (sub-)factors into classes, but the classes' rating is based on Roman numbers. Besides the three-level rating of the GAMP method, the rating of (sub-)factors is based on Arabic numerals. Thus, all low Arabic number ratings indicate low risk and vice versa. In addition, the ranges of the (sub-)factors' rating are more flexible since some (sub-)factors might need only two or more than three levels.
- *Step 3: Snowballing to extend the factor classification.* Related aspects of identified sub-factor are explored. For example, different *Change Type* of control code results in different change of comment which has different quality (i.e. *Comment Quality*). Thus, standards of comment (i.e. *Comment Compliance*) are included in the factor classification.
- *Step 4: Propose calculation rules for (sub-)factors.* Once sub-factors are selected, the calculation rules are proposed.
- *Step 5: Conduct multi-stage assessments.* The GAMP method starts with two sub-factors at level 2 (i.e. *Severity* and *Probability*) to assess their corresponding factor at level 1 (i.e. *Risk class*). Thus, the assessment might start from the lowest-level sub-factors and gradually move up the factor hierarchy. However, as mentioned, to enable flexibility, the calculation process and operators are free of choice.
- *Step 6: Visualise and interpret the results for further action.* The value of RPI4DD is reviewed to indicate the areas if the document is insufficient, incomplete, or outdated, which indicates documentation debt.

In the next section, the results of the approach are presented.

4.3 Factor classification of RPI4DD formulation

From the approach presented in Section 4.2, a basic structure to calculate RPI4DD is derived with two main influencing factors (cf. equation (3)). The *Document Urgency* factor

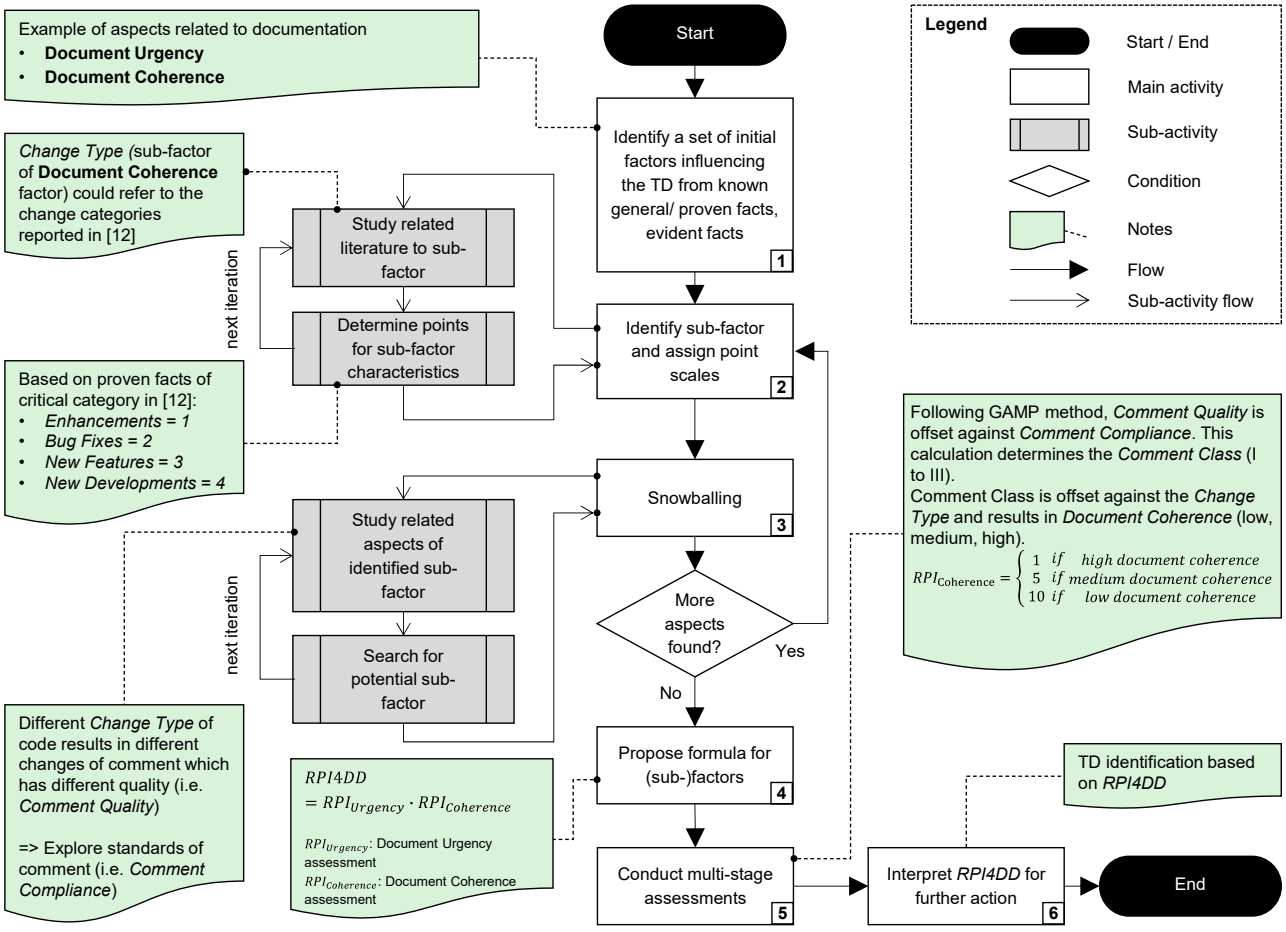


Figure 3: Steps to establish risk prioritisation indicator of documentation debt RPI4DD.

(represented by $RPI_{Urgency}$) assesses how acute the need for control code documentation is. $RPI_{Urgency}$ is based on the assumption that the urgency can vary depending on the *Functionality* ($RPI_{Functionality}$) implemented by the respective control code part, the *Required Change On-site* ($RPI_{On\text{-}site}$), as well as the *Grade of Test/Quality Of Test* (RPI_{Test}). The *Document Coherence* factor ($RPI_{Coherence}$) assesses how coherent the existing documentation is. The expert responses reported a sub-optimal standardisation of programming and documentation guidelines, i.e., company-specific guidelines are more frequently used than the guidelines provided by the community (cf. Section 3). Thus, a conformity check of documentation with standards shall take place. A summary of the groups of sub-factors influencing the control code documentation is presented in Figure 4.

$$RPI4DD = \underbrace{RPI_{Funtionality} \cdot RPI_{On-site} \cdot RPI_{Test}}_{RPI_{Urgency}} \cdot RPI_{Coherence} \tag{3}$$

In the following, the specification of sub-factors and corresponding rationale are described in the order presented in Figure 4.

4.3.1 Document urgency (RPI_{Urgency})

RPI_{Urgency} is calculated based on three sub-factors: (1) *Functionality*, (2) *Required Change On-site*, and (3) *Grade Of Test/Quality Of Test*.

4.3.1.1 Functionality ($RPI_{\text{Functionality}}$)

Depending on the complexity of an implemented functionality, different amounts of explanatory documentation are required to enable maintainability by different persons [26]. Poor functional description availability (#72) poses a high risk since comprehensibility (readability) received the highest rank among the software properties influenced by complexity [5]. Therefore, *Complexity Of Software* is identified as a sub-factor of *Functionality*. The *Overall Complexity* metric

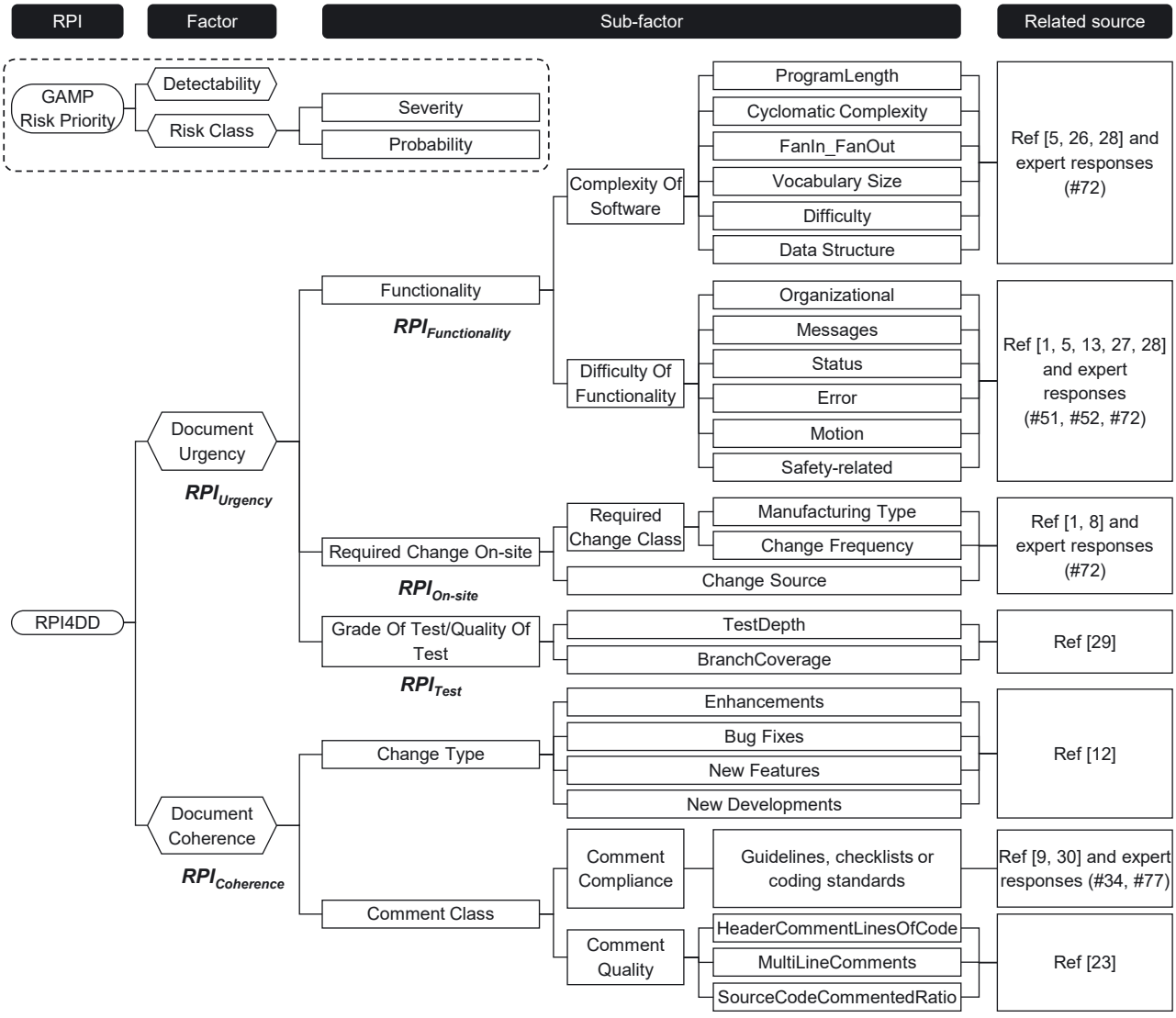


Figure 4: Selected factors influencing control code documentation (i.e. factor classification of RPI4DD formulation) following the systematic classification of Li et al. [15] and GAMP procedure [10].

proposed in [5] has proven as a reliable means to quantify different aspects of software complexity and, thus, could be used as a calculation rule for the sub-factor *Complexity Of Software*. As identified in [5], the list of characteristics includes:

- ProgramLength M_1
- Cyclomatic Complexity M_2
- FanIn_FanOut M_3
- Vocabulary Size M_4
- Difficulty M_5
- Data Structure M_6

It is worth noting that *Cyclomatic Complexity* or *Data Structure* may be weighted more strongly as they might have larger effects on the program complexity.

Second, the difficulty of software functionalities might vary [5]. As documentation activity is quite laborious and tedious, the documentation effort needs to be prioritised for the most difficult functionality (i.e., goal-oriented behavior [13]) of software. Therefore, *Difficulty Of Functionality* is included as another sub-factor of $RPI_{\text{Functionality}}$. As identified in [5, 13, 27], the list of functionalities is as follows.

- Organizational (low weight = 1).
- Messages (low weight = 1).

- Status (low weight = 1).
- Error/generic/homing (medium weight = 2)
- Motion Control (high weight = 3)
- Safety-related (high weight = 3).

Motion control in aPS is highly challenging in case a production step (e.g., work piece handling) requires the synchronisation of multiple drives performing different motion tasks (e.g., rotatory or linear movements [28]) by simultaneously fulfilling hard real-time requirements [27]. According to [1], positioning tasks of the robot-like-systems components are rated as one of the most challenging tasks to implement in software. Changes in production (e.g., introduction of a new product) might require additional or more complex movement; thus, the motion control must be adapted accordingly (e.g., changing the number of involved cooperating motion axes of robot arms). The difficulty of motion control is therefore rated as high (2). Safety-related tasks are even more critical since errors or malfunctions in these software parts may cause severe damage to aPS, or – in the worst case – even to humans; thus, they are rated with 3. Here, poor automatic control code generation (#51 and #52) might pose a high risk, e.g., safety-related control code might differ from the predefined model or requirements document.

Following the calculation rules presented in [5, 28] for POU complexity, a calculation for $RPI_{\text{Functionality}}$ is proposed as follows. As the medians $\tilde{M}_1$ to $\tilde{M}_6$ are used (cf. Section 2.2), they are "stable against outliers and provide reliable results" [5]. Therefore, the distribution of Overall Complexity OC values would be mostly symmetrical without outliers. Thus, $RPI_{\text{Functionality}}$ is based on the mean Overall Complexity $\overline{OC}$.

$$\begin{aligned} RPI_{\text{Functionality}} = {} & \overline{OC_{\text{Organizational}}} + \overline{OC_{\text{Messages}}} \\ & + \overline{OC_{\text{Status}}} + 2 \cdot \overline{OC_{\text{Error}}} \\ & + 2 \cdot \overline{OC_{\text{Motion}}} + 3 \cdot \overline{OC_{\text{Safety}}} \end{aligned} \tag{4}$$

4.3.1.2 Required change on-site ($RPI_{\text{On-site}}$)

The sub-factors of *Required Change On-site* include *Manufacturing Type, Change Frequency* and *Change Source*. First, while MM companies can usually develop machines entirely in-house, PM companies often have to integrate and start up the system on-site [8]. Thus, the *Manufacturing Type* is identified as a sub-factor for *Required Change On-site*. Second, customers have different schedules (e.g., new product introductions); therefore, the remote software update intervals (i.e., *Change Frequency*) vary. Third, while in-house changes often undergo a rigorous review process by the software development department and management, on-site changes are often conducted without feedback from the development department (e.g., due to time pressure). Therefore, the qualification of the person conducting a change or the origin of changes (*Change Source*) is identified as another sub-factor. Other potential sub-factors include code-sharing strategies, acquiring software status, or company sizes (documentation availability might vary in sizes of companies, according to #72).

Regarding the *Manufacturing Type*, the highest document urgency is rated at PM (rating = 3) as PM has the most required change on site. As on-site changes must be quickly performed (e.g., to meet hard deadlines or to reduce the cost of suspending production), it is urgent to provide exact instructions. The rating is lower for series machinery manufacturing (rating = 1) and special-purpose machinery manufacturing (rating = 2).

Regarding *Change Frequency*, the rating scale is based on the frequency of software updates reported in [1]:

- <3 months = 1
- 3–6 months = 2
- 6–12 months = 3
- > 12 months = 4
- No update = 5

Regarding *Change Source,* a rating of 1 is assigned to *Changes conducted in-house* (less risky due to rigorous review*),* and *On-site changes* are given a rating of 2.

The assessment on *Required Change On-site* is described in Figure 5. Following the idea in the GAMP method, this assessment includes two steps with three aspects. Firstly, the *Manufacturing Type* (severity of change) is plotted against the *Change Frequency* (probability that a change will occur), giving a *Required Change Class* (I to III). Secondly, the *Required Change Class* is plotted against *Change Source*, thus resulting in giving a degree of *Required Change On-site* (low, medium, high). There is an adaption in this assessment that the detectability aspect introduced in the GAMP method is replaced by *Change Source*. Nevertheless, the detectability concept is still included since the detectability of inadequate documentation may vary with different *Change Sources,* which might undergo different review processes.

Finally, the value $RPI_{\text{On-site}}$ is determined using equation (5).

$$RPI_{\text{On-site}} = \begin{cases} 1 & \text{if low urgency on-site} \\ 5 & \text{if medium urgency on-site} \\ 10 & \text{if high urgency on-site} \end{cases} \tag{5}$$

4.3.1.3 Grade Of Test/Quality Of Test (RPI_{Test})

In the aPS domain, the method proposed by Ulewicz et al. [29] can be followed to identify which control code snippets

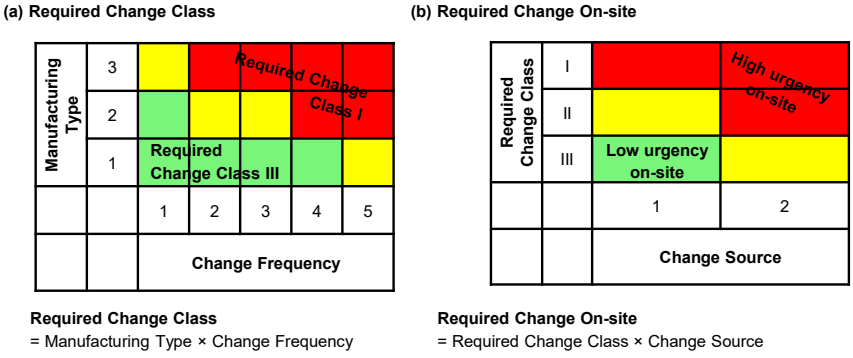


Figure 5: Required Change On-site assessment according to the GAMP method [10] (a: Required Change class calculation; b: Required Change On-site calculation).

are not well tested yet. The process employed the control code coverage method to determine which control code snippets are covered by which test cases. Among the proposed metrics in [29], there is a test coverage measure on the POU hierarchy (i.e., *TestDepth*) and three control code coverage measures (i.e., *BranchCoverage, StatementCoverage,* and *PathCoverage*). With *StatementCoverage*, it is aimed to traverse all the statements (or lines) at least once. *BranchCoverage* aims to traverse all the control flow paths (e.g., if statements) at least once. The main goal of *PathCoverage* is to execute all possible control flow paths (full paths from input to output). A brief comparison of these control code coverage metrics is illustrated in Figure 6. With test cases TC1 and TC2 (cf. Figure 6a), *StatementCoverage* shows 100 %; however, only 75 % of the branches are covered (*BranchCoverage*). Thus, *BranchCoverage* is a better indicator as it shows that more test cases are required. Regarding *PathCoverage*, there is a case that the number of paths through the loop could be significantly large (cf. Figure 6b), infinite, or not predictable (e.g., number of loops is determined at run time). Thus, covering all possible paths from input to output is mostly not realisable in practice. Therefore, among these control code coverage metrics, *BranchCoverage* is recommended as it delivers more accurate results.

In practice, 80 % coverage is generally accepted as good (following the Pareto principle in testing: given that 80 % of code is covered, the remaining code part may require a significant effort but is not worth it). To distinguish between a poor and a moderate test quality, additional information needs to be taken into account. More precisely, the expectation for test quality depends on various company-specific boundary conditions, such as software modularity or reusability. For instance, the boundary conditions at the industrial partner in the previous study [25] are as follows: the company has a high-modularised structure of its control code and a high degree of reuse by applying library modules and templates (about 75 % of a machine's PLC project consist of reused control code). The other 25 % of the machine's PLC project is newly developed machine- or customer-specific control code, e.g., data exchange between the reused library modules or implementation of the machine-specific process logic. These newly developed, machine-specific parts are the error-prone parts of the PLC project, which require thorough testing. Generally, the company's engineers know which requirements target these new code parts, and due to the company's mature code structure, they can locate the code parts in the PLC project. Consequently, the defined tests usually aim at targeting specifically the requirements linked to these critical code parts. Again, following the Pareto principle, at least 80 % of the 25 % new machine-specific code needs to be covered by tests. Since the 80 % of the 25 % new machine-specific code is about 20 % of the total code, the company-specific threshold for moderate test quality is set to 20 % of the entire control code in the considered PLC project. The threshold of 20 % for "moderate" is set given the assumption that test cases for the machine-specific part are selected and that the remaining code is already well-tested and thus assumed to be less error-prone. Therefore, a proposal to assess the test quality with two metrics, TestDepth and BranchCoverage, is shown in equation (6).

$$\text{TestQuality} = \begin{cases} \text{good} & \text{if TestDepth} \geq 80\% \wedge \text{BranchCoverage} \geq 80\% \\ \text{moderate} & \text{if } 20\% \leq \text{TestDepth, BranchCoverage} < 80\% \\ \text{poor} & \text{if TestDepth, BranchCoverage} < 20\% \end{cases} \tag{6}$$

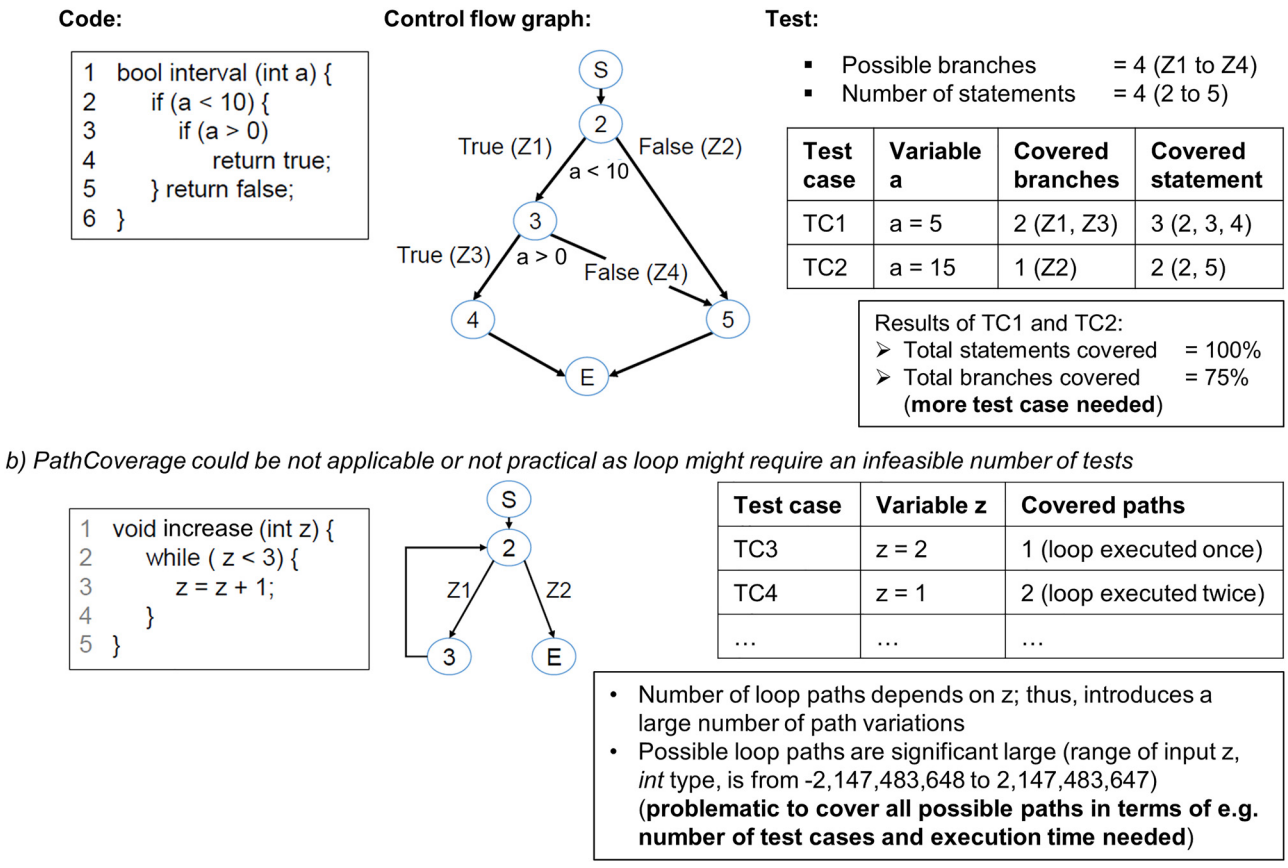


Figure 6: Comparison on control code coverage metrics proposed in [29] (a: Branch and Statement; b: Path).

Finally, the value RPI_{Test} is determined using equation (7).

$$RPI_{\text{Test}} = \begin{cases} 1 & \text{if good test quality} \\ 5 & \text{if moderate test quality} \\ 10 & \text{if poor test quality} \end{cases} \quad (7)$$

As this paper's scope focuses on control code documentation, there is no claim for completeness regarding test metrics. Nevertheless, many test metrics are available in the literature that can be integrated into the RPI_{Test} calculation.

4.3.2 Document Coherence ($RPI_{\text{Coherence}}$)

The *Document Coherence* factor is assessed based on the three sub-factors *Comment Compliance*, *Comment Quality* and *Change Type*. Firstly, practice shows that implementation of the standards could reduce errors. Violation of coding standards (e.g., MISRA) may result in TD [30]. A lack of exact instructions (#77) or a lack of coding guidelines in general (#34) might pose a risk, e.g., poor quality or standardised code. Therefore, *Comment Compliance* is identified as a sub-factor, which is categorised into the *Document Coherence* factor. Secondly, the coherence of control code documentation and corresponding implementation may influence the software's maintainability. Insufficient *Comment Quality* (e.g., code comments do not describe the implementation properly) may hinder on comprehensibility (readability) of the control code (e.g., causing confusion for the maintenance staff). Thirdly, different change categories (i.e., Enhancements, Bug Fixes, New Features, and New Developments) could influence the software maturity value, according to [12]. Thus, *Change Type* is identified as a factor within the *Document Coherence* factor.

Regarding *Comment Compliance*, the rating is based on results from checking the control code documentation with MISRA guidelines. MISRA rules or directives for comments include:

- *Rule 3.1 The character sequences /* and // shall not be used within a comment.*
- *Rule 3.2 Line-splicing shall not be used in // comments.*
- *Rule 20.1 #include directives should only be preceded by preprocessor directives or comments.*
- *Dir 4.4 Sections of code should not be "commented out"*

The details of the rating scale for *Comment Compliance* are presented in Table 5. Among the available guidelines, MISRA is proposed as its scope focuses on coding standards. As an alternative or a broader scope, one could refer to other guidelines or standards such as PLCopen, ISO 26262 or ISO 17961.

Regarding *Comment Quality (CQ)* rating, it is assumed that a high amount of comments is beneficial to enhance the software's understandability, and, thus, reduces the risk of documentation debt. The metrics *HeaderCommentLinesOfCode* (size of comment header), *MultiLineComments* (amount of comments wrapping over multiple lines), and *SourceCodeCommentedRatio* (density of comments) proposed in [23] could be employed to quantify the amount of control code comments (cf. equation (8)).

$$CQ = \begin{cases} 1 \text{ if } \text{SourceCodeCommentedRatio} > 0 \ \wedge \text{MultiLineComments} > 0 \wedge \text{HeaderCommentLinesOfCode} > 0 \\ 2 \text{ if } \text{SourceCodeCommentedRatio} > 0 \ \wedge (\text{MultiLineComments} = 0 \vee \text{HeaderCommentLinesOfCode} = 0) \\ 3 \text{ if } \text{SourceCodeCommentedRatio} = 0 \end{cases} \quad (8)$$

Regarding *Change Type*, the scale refers to the change categories reported in [12]. The details of the rating scale are presented in Table 6.

The *Document Coherence* assessment method is described in Figure 7. Following the idea of the GAMP method, firstly, the *Comment Quality* (determined using equation (8)) is offset against the *Comment Compliance* (determined using the rating scale in Table 5). This calculation determines the degree of excellence of comment, i.e., *Comment Class* (I to III). Secondly, the *Comment Class* is offset against the degree of excellence of control code (represented by *Change Type*, which influences the software maturity value). It results in *Document Coherence* (low, medium, high). The *Change Type* is determined using the rating scale in Table 6. It could be observed that the three aspects defined in GAMP (i.e., severity, probability, and detectability) are not used but replaced by the three new aspects in the context of coherent assessment.

Finally, the $RPI_{\text{Coherence}}$ is determined with equation (9).

$$RPI_{\text{Coherence}} = \begin{cases} 1 \text{ if high document coherence} \\ 5 \text{ if medium document coherence} \\ 10 \text{ if low document coherence} \end{cases} \quad (9)$$

4.4 Proof of concept of risk prioritisation indicator of documentation debt

To illustrate the applicability, RPI4DD is in the following determined for two scenarios of the extended Pick and Place Unit (xPPU) [8], i.e., a lab-sized demonstrator that stamps, transports, and sorts work pieces with different color, weight, and material (cf. Figure 8). The work pieces arrive at the *stack*, where they are picked up by the *crane*. Depending on the work piece type, they are either further processed to the *stamp* or directly transported with the *conveyor* to be sorted into the respective *ramp*.

In the following, RPI4DD is calculated for two typical functionalities in aPS emulated for the xPPU, i.e.:

- *F_Stamp: Stamping work pieces:* This scenario refers to the code parts controlling a typical functionality of the system during regular production, i.e., in the case of the xPPU, the stamping of work pieces at the crane before they are sorted.
- *F_Restart: Restart operation after emergency stop:* The recovery of the system after an emergency stop to take up regular production is a highly challenging task in industrial practice, e.g., due to complex resynchronisation of drives, which is why the scenario is considered on a small scale for the xPPU.

Table 7 follows a practical risk assessment template used to evaluate risks during the development of industrial aPS. The table template can serve different views to different stakeholders. The individual sub-factors are intended to be initially determined by technicians on a fine-grained level resulting in precise numbers for the different factors (e.g., results from complexity metrics). Next, the fine-grained values are clustered and categorised into a coarse-grained system (e.g., following a three-level categorisation into low,

Table 5: Rating scale for Comment Compliance with guidelines or standards.

Rating	Description	Comment Compliance from checking with guidelines (e.g., MISRA)
1	Noncritical	Fully compliance (i.e., no error or warning messages)
2	Minor	Partially compliance (i.e., violations only include warning messages)
3	Critical	Noncompliance (i.e., violations include error messages)

Table 6: Rating scale for Change Type based on scope and criticality.

Rating	Description	Rationale according to [12]
1	Enhancements	*[…] represent the* ***least critical*** *category […]*
2	Bug fixes	*[…] represent a* ***more critical*** *[…]* ***than enhancements*** *[…]*
3	New features	*[…]* ***significantly more critical****, […]*
4	New developments	*[…]* ***the most extensive change category*** *[…]*

The bold terms highlight the exact phases extracted from ref. [12] explaining the rationale for the corresponding ratings in column 1.

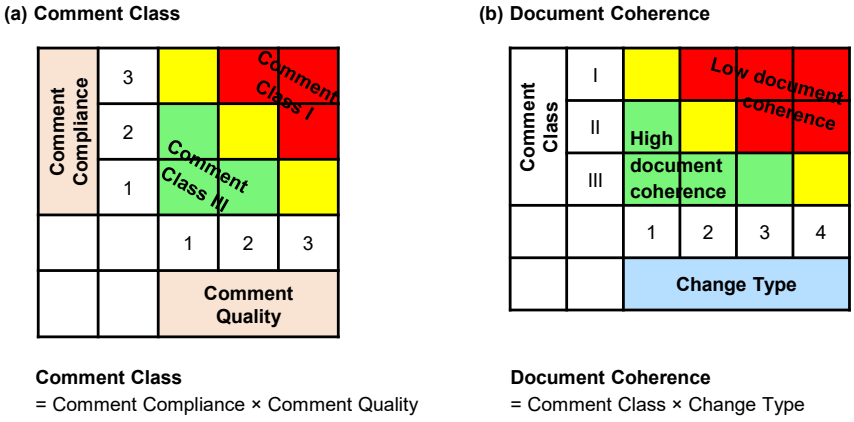


Figure 7: Document Coherence assessment according to the GAMP method [10] (a: Comment Class calculation; b: Document Coherence calculation).

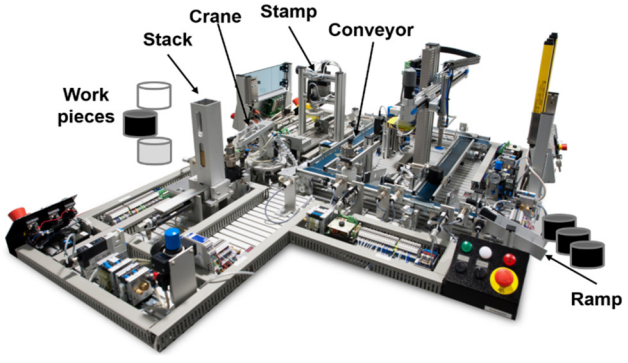


Figure 8: Extended Pick and Place Unit [8] to apply the risk priority indicator RPI4DD.

medium, and high) to make the risk priority number intuitively understandable at first glance, e.g., for customers or management positions.

In the following, the sub-factors are determined for both scenarios to derive RPI4DD for the software parts implementing the respective functionalities.

4.4.1 Calculation of document urgency (RPI_{Urgency})

Regarding the *Functionality*, F_Restart might involve human intervention and multiple safety-related tasks; therefore, it is assumed that documentation for F_Restart is more critical than documentation for F_Stamp. Particularly, the *Complexity Of Software* and *Difficulty Of Functionality* of POUs in F_Restart is assumed to be higher than F_Stamp's in general; thus, $RPI_{\text{Functionality}}$ of F_Restart appears to be higher than $RPI_{\text{Functionality}}$ of F_ Stamp, according to $RPI_{\text{Functionality}}$ calculation rules in Section 4.3. Hence, $RPI_{\text{Functionality}}$ of F_ Stamp and F_Restart are rated with *5* and *10*, respectively. As $RPI_{\text{Functionality}}$ calculation rules follow the mechanism of [5], they are fully automatable (cf. **requirement R3**).

Table 7: A (fictive) example on RPI4DD calculation on two xPPU functionalities Stamping work pieces (F_Stamp) and Restart operation after emergency stop (F_Restart).

		Functionality	
		F_Stamp	**F_Restart**
Document Urgency	**Functionality** $RPI_{Functionality}$	5	10
	Required Change On-site $RPI_{On\text{-}site}$	1	5
	Grade Of Test/Quality Of Test RPI_{Test}	10	5
	$RPI_{Urgency}$ ($RPI_{Functionality}$ x $RPI_{On\text{-}site}$ x RPI_{Test})	50	250
Document Coherence (*)	**Comment Compliance**	3	3
	Comment Quality	1	2
	Comment Class (Comment Compliance x Comment Quality)	3	6
		II	I
	Change Type	1	3
	$RPI_{Coherence}$ (Comment Class x Change Type)	3	18
		1	10
RPI4DD ($RPI_{Urgency}$ x $RPI_{Coherence}$) (**)		50	2500
		Low	High

(*) cf. Fig. 7 and equation (9) for calculation rules of Document Coherence

(**) Risk Priority Risk of Documentation is assumed

- Low if RPI4DD ≤ 625 (5x5x5x5)
- Medium if 625 < RPI4DD < 1250 (10x5x5x5)
- High if RPI4DD ≥ 1250

([a]) cf. Figure 7 and equation (9) for calculation rules of Document Coherence. ([b]) Risk Priority Risk of Documentation is assumed. Low if RPI4DD ≤ 625 ($5 \times 5 \times 5 \times 5$). Medium if $625 <$ RPI4DD < 1250 ($10 \times 5 \times 5 \times 5$). High if RPI4DD ≥ 1250.

Regarding the *Required Change On-site*, due to the involvement of human intervention and multiple tasks, it is assumed that F_Restart has a larger scope and requires more on-site changes than F_Stamp. In particular, due to multiple-machine involvement, the F_Restart would mostly need to be integrated on-site by PM, while F_Stamp can be seen as a series machine which could be developed in-house by MM. Thus, the *Manufacturing Type* of F_Restart tends to be larger than F_Stamp. The same tendency applies to *Change Frequency* and *Change Source*. Thus, $RPI_{On\text{-}site}$ of F_Restart appears to be higher than $RPI_{On\text{-}site}$ of F_Stamp, according to the $RPI_{On\text{-}site}$ calculation rules listed in Section 4.3. Hence, the *Required Change On-site* of F_Stamp and F_Restart are rated with *1* and *5*, respectively. It should be noted that the calculation structure is extensible (cf. **requirement R1**), which allows the integration of other sub-factors. For instance, *Facility Availability* sub-factor can be included to present the readiness of required equipment to conduct the changes. In addition, the listed rating scales of *Manufacturing Type*, *Change Frequency* and *Change Source* can be modified (cf. **requirement R2**). For instance, a new item *Changes conducted both in-house and on-site* can be included in *Change Source* for a large project.

Regarding the *Grade Of Test/Quality Of Test*, per larger scope assumption, F_Restart has larger impact than F_Stamp in the whole aPS; therefore, it is assumed that more resources are allocated on testing and documentation for F_Restart. It is because a defective stamping machine could be quickly fixed or replaced to resume operation, while a suspension of the whole production to fix an issue in F_Restart might result in a high cost to both aPS manufacturer and customers. With larger test efforts spent, *Test-Depth* and *BranchCoverage* of F_Restart are assumed to be larger than 80 % (good test quality) and F_Stamp's are below

80 % (moderate). Thus, RPI_{Test} of F_Restart appears to be less than RPI_{Test} of F_Stamp, according to the equations (6) and (7). Hence, RPI_{Test} of F_Stamp and F_Restart are rated as *10* and *5*, respectively. As results of the test metrics are provided by test management tools [29], the *Grade Of Test/Quality Of Test* calculation can be fully automatable (cf. **requirement R3**).

The RPI_{Urgency} is determined by multiplying the $RPI_{\text{Functionality}}$, $RPI_{\text{On-site}}$ and RPI_{Test}, according to equation (3).

4.4.2 Calculation of Document Coherence ($RPI_{\text{Coherence}}$)

Regarding the *Comment Compliance*, the scopes of both functionalities are not trivial; therefore, it is assumed that there are error messages associated with the *Comment Compliance* rules listed in Section 4.3. For the fictive examples focused for the *RPI* calculation for the xPPU, it is assumed that both contain comments that are non-compliant with the MISRA comment guidelines, e.g., in the implementation of the crane (cf. Figure 9). These non-compliant violations are critical for both functionalities F_Stamp and F_Restart, thus are rated with *3* for *Comment Compliance.* It should be noted that there are various configurable tools are available to check compliance with MISRA rules, enabling automated assessment for this criterion (cf. **requirement R3**).

Regarding *Comment Quality*, as aforementioned, F_Restart scope is larger and relates to plant manufacturing which mostly requires on-site changes, while F_Stamp scope mostly relates to series machine manufacturing which could be mainly developed in-house. Therefore, it is assumed that the three metrics *HeaderCommentLinesOfCode*, *MultiLineComments*, and *SourceCodeCommentedRatio* deliver better results at F_Stamp than F_Restart. For instance, some header comments in F_Restart on-site changes might be neglected due to high time pressure and short deadline, resulting in HeaderCommentLinesOfCode $= 0$. Thus, the *Comment Quality* of F_Restart appears to be larger than F_Stamp, according to calculation rules in equation (8). Hence, the *Comment Quality* of F_Stamp and F_Restart are rated as *1* and *2*, respectively.

The *Comment Class* is determined by offsetting the *Comment Compliance* against *Comment Quality* (cf. detailed calculation rules in Figure 7a).

Regarding the *Change Type*, a change at F_Restart often requires a configuration of multiple tasks at multiple components; therefore, it is assumed that a change at F_Restart is generally larger than a change at F_Stamp (e.g., mainly New Features at F_Restart and mainly Enhancements at F_Stamp). Thus, according to the rating scale in Table 6, *Change Type* of F_Stamp and F_Restart are rated with *1* and *3*, respectively.

Altogether, the RPI4DD is determined by multiplying the RPI_{Urgency} and $RPI_{\text{Coherence}}$, according to equation (3). The RPI4DD results indicate that the risk of documentation debt at F_Restart (2500) is higher than at F_Stamp (50). Thus, document improvements or actions to prevent damage from documentation TD related to F_Restart should be planned for corresponding module developers or application engineers. When transferring the results received for the xPPU to real-world production systems, companies may benefit by RPI4DD as a quantitative indicator for different stakeholders (e.g., from management or software development) to prioritize starting points for reducing the amount of documentation debt and, thus, avoid high long-term cost.

4.5 Summary of RPI4DD calculation and requirement fulfilment

The calculation for RPI4DD is summarised as follows (cf. equation (10)). The range of each factor $RPI_{\text{Functionality}}$, $RPI_{\text{On-site}}$, RPI_{Test} and $RPI_{\text{Coherence}}$ is *(0, 10]* with an assumption that all operands are equally critical for TD. This proposal employs the widely-used scale up to *10* for these factors to simplify the calculations. To scale the *RPI* to another range, one can use weights

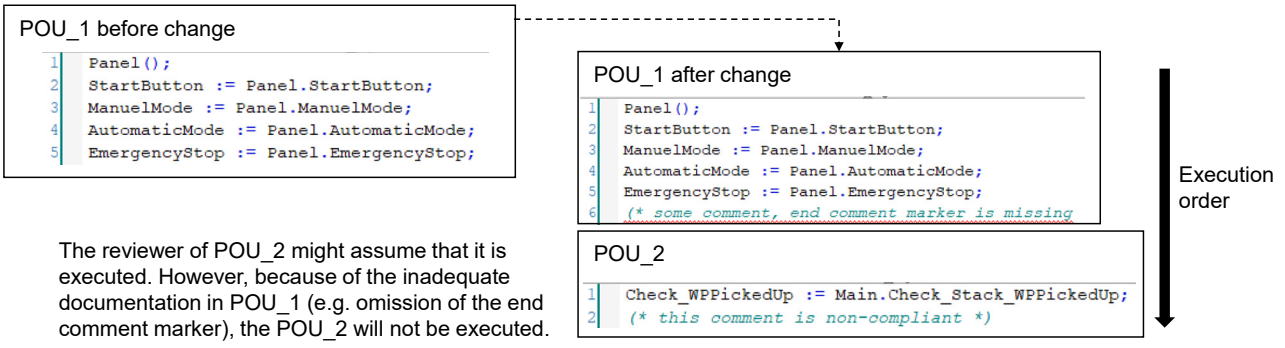


Figure 9: A (fictive) example of non-compliant comment based on MISRA rule 3.1 example [9] and xPPU [8].

(cf. $RPI_{\text{Functionality}}$ calculation) or offset the factors (cf. $RPI_{\text{On-site}}$ calculation). It could be observed that the range of RPI_{Urgency} (multiplication of $RPI_{\text{Functionality}}$, $RPI_{\text{On-site}}$ and RPI_{Test}) (0, 1000] is larger than $RPI_{\text{Coherence}}$ range (0, 10]. It is due to the criticality of documentation being considered more important than the coherence of documentation. For example, if a commissioner has to make changes at the construction site at the customer's site, it is crucial that documentation is available (urgent), but it is not a priority whether the comments fulfil the MISRA guidelines (coherent). It is worth noting that the factor *Grade Of Test/Quality Of Test* could be optional. RPI_{Test} is only relevant in case changes are conducted in-house (e.g., by module developer or application engineer), and test results are available for assessment. RPI_{Test} acts as a considered issue to tweak RPI4DD. Square brackets denote the optionality of RPI_{Test}.

$$\begin{aligned} \text{RPI4DD} &\cong RPI_{\text{Urgency}} \cdot RPI_{\text{Coherence}} \\ &\cong \left(RPI_{\text{Functionality}} \cdot RPI_{\text{On-site}}\left[\cdot RPI_{\text{Test}}\right]\right) \\ &\quad \cdot RPI_{\text{Coherence}} \end{aligned} \tag{10}$$

The RPI4DD approach is achieved with a set of selected factors to indicate control code documentation debt in aPS (cf. Figure 4), which is extensible *(R1)*, flexible *(R2)*, and automatable *(R3),* therefore, feasible for industrial large-scale software projects, which often involve hundreds of POUs. The extensibility and flexibility characteristics presented in the *Required Change On-site* assessment (cf. Section 4.4) can generally be applied to assessments of other aspects (cf. *Facility Availability* and *Manufacturing Type* examples in *Required Change On-site* in Section 4.4). New sub-factors can be systematically included in the proposed hierarchy (cf. Figure 4). All proposed rating scales or calculation rules are adjustable. Thus, ***R1 (extensible)*** and ***R2 (flexible)*** are satisfied. ***R3 (automatable)*** is evaluated as partially fulfilled since all presented calculation rules can be automated except *Change Source*, which might require a manual assessment. Nevertheless, with trivial effort, semi-automatic assessment methods can be developed to classify *Change Source* (e.g., by a configuration file).

To summarise, the integration of RPI4DD into the risk management process of machine and plant manufacturing companies yields high potential to support practitioners in identifying and analysing the risks associated with documentation debt in technical systems by providing an intuitive quantitative indicator based on established risk assessment approaches. Therefore, earlier reaction or risk-reduction measures can be developed to enhance the system quality and drastically reduce costs by preventing errors and mal-functions caused by a lack of documentation in early phases.

5 Conclusion and outlook

The current state of practice regarding documentation in aPS manufacturing companies clearly showed a lack of documentation in industrial development, especially in control software, causing a high risk of maintainability issues in later lifecycle phases and thus, increased cost. In particular, the evaluation of expert surveys indicated low document accessibility, especially in early engineering phases, poor automatically generated engineering artefacts, inadequate formulation of guidelines, and highly dependent on in-house guidelines in MM and PM. To address this issue, a Risk Prioritisation Indicator of Documentation Debt (RPI4DD) is proposed to quantify the risk of documentation debt in control software. Related work addressing the quality of control code documentation is still scarce and primarily considers source code alone. To the best of the authors' knowledge, this is the first study to transfer the Risk Priority concept in the GAMP method to automation software and its boundary conditions. The RPI4DD approach considers not only internal software factors (e.g., complexity or functionalities) but also includes external influencing factors, such as required on-site changes or tests. Therefore, compared to existing work, the proposed approach could provide a broader view on control code documentation since aPS is developed in a multidisciplinary environment. Besides documentation, the proposed approach could be generally applied to other aspects of software quality (e.g., testing) or could be further applied to other disciplines.

The applicability of the approach is evaluated on a lab-sized demonstrator. Based on the risk priorities obtained, follow-up documentation activities can be determined. No action is required in case the risk priority is low, e.g., F_Stamp in the proof of concept presented in Section 4.4. If the risk is high, e.g., F_Restart, the factors contributing most to the risk must be analysed to plan for additional documentation tasks, e.g., to review and note changes of F_Restart at commissioning since F_Restart would mostly need to be integrated on-site by PM – as aforementioned. If the outcome is a medium level of risk, it is necessary to check if current documentation is sufficient for staff working with the control code.

The presented factors covering an initial basis of influences need to be adapted or enlarged (e.g., quality of code comments is not yet covered). The evaluated use case is limited to the scope of a lab-sized demonstrator consisting of general functionalities. There may be unexpected

company-specific software structure and documentation. More concrete factors might be required since the code size of industrial applications may vary from a hundred to several thousand lines of code per POU [12]. In future work, it is therefore planned to evaluate RPI4DD in industrial settings to enhance the factor hierarchy and calculation rules. First, the workflow with involved stakeholders and communication interfaces between departments would need a study to collect the required information for RPI4DD calculation in practice. Second, control code documentation refers not only to code comments but also to manuals or architect documents; thus, these documentation artefacts should be included in future studies to further develop the factor classification for the extensible RPI4DD. Furthermore, new tools can be integrated or designed to enable automatic assessments for the identified factors to promote the RPI4DD concept to a framework applicable to industrial development workflows.

Acknowledgements: We would like to thank Kathrin Land and Juliane Fischer for the support in the research. We would also thank the Vietnam International Education Development and Vietnamese-German University for granting Quang Huan Dong a scholarship under Project 911 – "Training lecturers of Doctor's Degree for universities and colleges for the 2010–2020 period" (Decision No. 771/QD BGDDT dated 14/03/2017).
Author contributions: All the authors have accepted responsibility for the entire content of this submitted manuscript and approved submission.
Research funding: None declared.
Conflict of interest statement: The authors declare no conflicts of interest regarding this article.

Appendix

This section provides excerpt of the questionnaire used in this study. The full questionnaire is available online [24].

- #34: What guidelines/checklists does your company use to ensure high quality and traceability software?
- #51: What types of planned reuse do you use for the development of open-/closed-loop or drive software?
- #52: From which tools/models is code generated?
- #65: Which sales markets is your company active in?
- #66: How many employees work in your company?
- #69: Would you classify your company more as a machine or plant engineering company?
- #72: When are the functional descriptions available in the respective department?
- #77: Does your company use internal guidelines for the respective discipline that serve to achieve results of the highest possible quality and traceability?

References

[1] B. Vogel-Heuser and F. Ocker, "Maintainability and evolvability of control software in machine and plant manufacturing — an industrial survey," *Control Eng. Pract.*, vol. 80, pp. 157—173, 2018.
[2] A. Lüder, A. Klostermeyer, J. Peschke, A. Bratoukhine, and T. Sauter, "Distributed automation: pabadis versus hms," *IEEE Trans. Industr. Inform.*, vol. 1, no. 3, pp. 31—38, 2005.
[3] Q. H. Dong and B. Vogel-Heuser, "Modelling technical compromises in electronics manufacturing with BPMN+TD — an industrial use case," in *IFAC-PapersOnLine: 17th IFAC Symposium on Information Control Problems in Manufacturing*, vol. 54, 2021, pp. 912—917.
[4] V. Vyatkin, "Software engineering in industrial automation: state-of-the-art review," *IEEE Trans. Industr. Inform.*, vol. 9, no. 3, pp. 1234—1249, 2013.
[5] J. Fischer, B. Vogel-Heuser, H. Schneider, N. Langer, M. Felger, and M. Bengel, "Measuring the overall complexity of graphical and textual iec 61131-3 control software," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 5784—5791, 2021.
[6] B. Vogel-Heuser, S. Rösch, A. Martini, and M. Tichy, "Technical debt in automated production systems," in *IEEE 7th Workshop on Managing Technical Debt*, 2015, pp. 49—52.
[7] B. Vogel-Heuser, E.-M. Neumann, and J. Fischer, "Maturity levels for automation software engineering in automated production systems," in *2022 IEEE International Conference on Industrial Informatics*, 2022, pp. 618—623.
[8] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: challenges and research directions," *J. Syst. Softw.*, vol. 110, pp. 54—84, 2015.
[9] MISRA, *MISRA*, [Online], Available at: www.misra.org.uk [accessed: Oct. 26, 2022].
[10] ISPE, *GAMP 5 Guide: Compliant GxP Computerized Systems*, [Online], Available at: https://ispe.org/publications/guidance-documents/gamp-5 [accessed: Oct. 26, 2022].
[11] T. Besker, A. Martini, and J. Bosch, "Software developer productivity loss due to technical debt—a replication and extension study examining developers' development work," *J. Syst. Softw.*, vol. 156, pp. 41—61, 2019.
[12] B. Vogel-Heuser, E. Neumann, and J. Fischer, "MICOSE4aPS: industrially applicable maturity metric to improve systematic reuse of control software," *ACM Trans. Softw. Eng. Methodol.*, vol. 31, no. 1, pp. 1—24, 2022.
[13] J. Wilch, J. Fischer, N. Langer, M. Felger, M. Bengel, and B. Vogel-Heuser, "Towards automatic generation of functionality semantics to improve PLC software modularisation," *At — Automatisierungstechnik*, vol. 70, no. 2, pp. 181—191, 2022.
[14] T. Besker, A. Martini, J. Bosch, and M. Tichy, "An investigation of technical debt in automatic production systems," in *Proceedings of the XP2017 Scientific Workshops*, 2017, pp. 1—7.
[15] Z. Li, P. Avgeriou, and P. Liang, "A systematic mapping study on technical debt and its management," *J. Syst. Softw.*, vol. 101, pp. 193—220, 2015.

[16] P. Avgeriou, P. Kruchten, I. Ozkaya, and C. Seaman, "Managing technical debt in software engineering (dagstuhl seminar 16162)," *Dagstuhl Rep.*, vol. 6, no. 4, pp. 110–138, 2016.
[17] T. Detofeno, A. Malucelli, and S. Reinehr, "PriorTD: a method for prioritization technical debt," in *Proceedings of the XXXVI Brazilian Symposium on Software Engineering*, 2022, pp. 230–240.
[18] A. Martini, J. Bosch, and M. Chaudron, "Investigating Architectural Technical Debt accumulation and refactoring over time: a multiple-case study," *Inf. Softw. Technol.*, vol. 67, pp. 237–253, 2015.
[19] S. Biffl, L. Kathrein, A. Lüder, et al., "Software engineering risks from technical debt in the representation of product/ion knowledge," in *Proceedings of the 31st International Conference on Software Engineering and Knowledge Engineering*, 2019, pp. 693–700.
[20] L. Waltersdorfer, F. Rinker, L. Kathrein, and S. Biffl, "Experiences with technical debt and management strategies in production systems engineering," in *Proceedings of the 3rd International Conference on Technical Debt*, 2020, pp. 41–50.
[21] Q. H. Dong, F. Ocker, and B. Vogel-Heuser, "Technical Debt as indicator for weaknesses in engineering of automated production systems," *Prod. Eng.*, vol. 13, nos. 3–4, pp. 273–282, 2019.
[22] E. M. Neumann, B. Vogel-Heuser, J. Fischer, S. Diehm, M. Schwarz, and T. Englert, "Automation software architectures in automated production systems: an industrial case study in the packaging machine industry," *Prod. Eng.*, vol. 16, pp. 847–856, 2022.
[23] E. M. Neumann, M. Gnadlinger, J. Fischer, L. Reimoser, S. Diehm, and M. Schwarz, "Metric-based identification of target conflicts in the development of industrial automation software libraries," in *2022 IEEE International Conference on Industrial Engineering and Engineering Management*, 2022. (*accepted*).
[24] B. Vogel-Heuser, J. Fischer, E.-M. Neumann, and M. Kreiner, *Success Factors For the Design of Field-Level Control Code in Machine and Plant Manufacturing — an Industrial Survey*, [Online], Available at: https://www.researchsquare.com/article/rs-168613/latest [accessed: Oct. 26, 2022].
[25] Q. H. Dong and B. Vogel-Heuser, *Including Validation of Process Control Systems' Engineering into the Technical Debt Classification*, Forschung im Ingenieurwesen/Engineering Research, Berlin, Springer–Verlag GmbH, 2022, (*submitted*).
[26] B. Vogel-Heuser, M. Obermeier, S. Braun, K. Sommer, F. Jobst, and K. Schweizer, "Evaluation of a UML-based versus an IEC 61131-3-based software engineering approach for teaching PLC programming," *IEEE Trans. Educ.*, vol. 56, no. 3, pp. 329–335, 2013.
[27] B. Vogel-Heuser, M. Zimmermann, K. Stahl, et al., "Current challenges in the design of drives for robot-like systems," in *Proceedings of 2020 IEEE International Conference on Systems*, Man, and Cybernetics, 2020, pp. 1923–1928.
[28] B. Vogel-Heuser, J. Fischer, D. Hess, E. M. Neumann, and M. Wurr, "Boosting extra-functional code reusability in cyber-physical production systems: the error handling case study," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 1, pp. 60–73, 2022.
[29] S. Ulewicz and B. Vogel-Heuser, "Industrially applicable system regression test prioritisation in production automation," *IEEE Trans. Autom.*, vol. 15, no. 4, pp. 1839–1851, 2018.
[30] A. Ampatzoglou, N. Mittas, A.-A. Tsintzira, et al., "Exploring the relation between technical debt principal and interest: an empirical approach," *Inf. Softw. Technol.*, vol. 128, p. 106391, 2020.

Bionotes

Quang Huan Dong
Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany
Vietnamese-German University, Thu Dau Mot City, Vietnam
huan.dong@tum.de

Quang Huan Dong is a PhD candidate at Institute of Automation and Information Systems at Technical University of Munich. His research interest is technical debt in automated production systems.

Birgit Vogel-Heuser
Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany
Core Member of MDSI and Lead of Sector Work of MIRMI, Technical University of Munich, Munich, Germany
Munich Institute of Integrated Materials, Energy and Process Engineering, Technical University of Munich, Garching, Germany
vogel-heuser@tum.de

Prof. Dr.-Ing. Birgit Vogel-Heuser received a Diploma degree in Electrical Engineering and a Ph. D. degree in Mechanical Engineering from RWTH Aachen. Since 2009, she is a full professor and director of the Insititute of Automation and Information Systems at the Technical University of Munich (TUM). Her current research focuses on systems and software engineering. She is member of the acatech (German National Academy of Science and Engineering), editor of IEEE T-ASE and member of the science board of MIRMI at TUM.

Eva-Maria Neumann
Institute of Automation and Information Systems, Department of Mechanical Engineering, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany
eva-maria.neumann@tum.de

Eva-Maria Neumann received an M.Sc. in Mechanical Engineering from Technical University of Munich (TUM), Munich, Germany in 2018. She is currently pursuing a Ph.D. at the Institute of Automation and Information Systems at TUM. Her main research interests are static code analysis and metrics to quantify control software quality and the function-oriented design of modular control software architectures to enhance its reusability.